SIAKE

基于超奇异同源的认证密钥交换协议*

第二轮算法说明

提交人：薛海洋，路献辉，王鲲鹏，田松，徐秀，贺婧楠，李宝

2019 年 10 月 20 日

*薛海洋由以下项目资助:国家自然科学基金(项目批准号:61602473)，“十三五”国家密码发展基金(课题号：MMJJ20170116)．路献辉由以下项目资助：国家自然科学基金(项目批准号:61572495)，“十三五”国家密码发展基金(课题号：MMJJ20170123)．李宝由国家自然科学基金(项目批准号:61772515)资助. 王鲲鹏由国家自然科学基金(项目批准号:61502487)资助. 田松由国家自然科学基金(项目批准号:61802401)资助.

摘要

SIAKE (Supersingular Isogeny based Authenticated Key Exchange)算法为基于超奇异椭圆曲线同源问题设计的隐式认证密钥交换协议，其主要技术来源于算法设计团队的三篇文章 [XLL+18, XXW+19, XAY+19]. 具体地，基于 [XLL+18]中提出的双密钥公钥加密算法，结合 [XXW+19]中提出的转换框架，我们设计了 2 轮的认证密钥交换协议 SIAKE 并证明其在经典随机预言模型下的安全性. 并最后在 [XAY+19] 中证明了其量子随机预言模型下的安全性.

我们在经典模型和量子模型下证明了算法的理论安全性. 在经典和量子随机预言模型下，基于判定 SIDH 假设，SIAKE 支持任意注册并满足 CK^+ 安全性. 即支持用户任意注册、满足弱前向安全性、抵抗 KCI 攻击、抵抗 MEX 攻击.

与 NIST后量子密码算法标准候选算法中唯一的超奇异同源算法 SIKE 相比，SIAKE 的优点是提供了认证性，并考虑了认证密钥交换协议中几乎是最强大的敌手攻击能力，提供了弱前向安全性、任意注册、抵抗 KCI 攻击和抵抗 MEX 攻击安全性; 可以提供量子随机预言模型下的安全性；具有极低的带宽通信量；由于算法的模块化设计特点，任意底层同源计算和曲线参数的改进和优化都可以应用到该算法中.

目录

1 基本知识与工具

本节介绍一些理解 SIAKE 所必须的基本的数学知识、David Jao 等人提交给 NIST 的 SIKE [JAC17] 中的基础密钥交换同源算法、超奇异同源相关的困难性假设、认证密钥交换协议的 CK^+ 安全模型，以及双密钥公钥加密方案的定义.

1.1 数学基础部分

1.1.1 有限域

域是可以进行四则运算的集合，而有限域是元素个数有限的域. 也就是说，有限域是在其上定义了加法和乘法两种运算的有限集合，而这个有限集合对加法成群，去掉加法的单位元 0 后的子集合对乘法成群，且乘法对加法有分配律.

有限素域是密码学中常用的有限域. 有限素域也是最简单的有限域，其元素个数为素数. 设 p 是一个素数. 我们记含 p 个元素的有限域为 $\mathbb{F}_p$，其通常的构造是用整数环 $\mathbb{Z}$ 模 $p\mathbb{Z}$ 得到的 p 元集合，其上有自然定义的加法和乘法，它们继承 $\mathbb{Z}$ 的加法和乘法并模 p 得到.

更复杂的有限域可以通过域扩张的方法得到. 有限域通过域扩张得到有限域的扩张过程一定是代数扩张. 设 $\mathbb{F}_q$ 是一个具有 q（不一定是素数）个元素的有限域，$f(X) \in \mathbb{F}_q[X]$ 是其上的一个 n 次不可约多项式，α 是它的一个根. 我们可以看到，用 α 代替未定元 X 得到的 $\mathbb{F}_q$ 上的多项式环 $\mathbb{F}_q[\alpha]$ 和仿照整数环模素数的方法，由 $\mathbb{F}_q[X]$ 模 $f(X)$ 得到的域同构，即有 $\mathbb{F}_{q^n} = \mathbb{F}_q[\alpha] \cong \mathbb{F}_q[X]/\langle f(X)\rangle$ 是一个域. 从这个构造方法可以看出，$\mathbb{F}_{q^n}$ 是有限域 $\mathbb{F}_q$ 上的一个 n 维线性空间，$1, \alpha, \alpha^2, \ldots, \alpha^{n-1}$ 是一组基.

一般来说，有限域 $\mathbb{F}_q$ 均包含一个有限素域为其子域，例如 $\mathbb{F}_q$ 的乘法单位元 1 的全体倍数构成一个有限素域 $\mathbb{F}_p$. 根据上面的论述，$\mathbb{F}_q$ 是 $\mathbb{F}_p$ 的扩域，因此其元素个数必为 p 的方幂，即有正整数 s，使得 $q = p^s$. 此时我们说域 $\mathbb{F}_q$ 的特征为 p，记为 $char(\mathbb{F}_q) = p$.

在利用超奇异椭圆曲线的同源运算构造的密码系统中，我们常常把椭圆曲线定义在一类特殊的 p^2 元有限域上. 这类有限域的特征是形为 $p = 2^{e_2}3^{e_3} - 1$ 的素数，其中 e_2，e_3 是正整数. 若 $e_2 \geq 2$，则 $p \equiv 3 \mod 4$，从而 -1 是模 p 的二次非剩余，因此 $X^2 + 1$ 是 $\mathbb{F}_p[X]$ 中的不可约多项式. 记它的一个根为 i，则有 $\mathbb{F}_{p^2} = \mathbb{F}_p[i] \cong \mathbb{F}_p[X]/\langle X^2 + 1\rangle$.

显然，$\mathbb{F}_{p^2}$ 中的元素可以表示为 $a + b \cdot i$，其中 $i^2 + 1 = 0$, $a, b \in \mathbb{F}_p$. $\mathbb{F}_{p^2}$ 上的加法公式为 $(a_1 + b_1 \cdot i) + (a_2 + b_2 \cdot i) = (a_1 + a_2) + (b_1 + b_2) \cdot i$，0 为加法单位元；乘法公式为 $(a_1 + b_1 \cdot i) \cdot (a_2 + b_2 \cdot i) = (a_1a_2 - b_1b_2) + (a_1b_2 + b_1a_2) \cdot i$，1为乘法单位元.

1.1.2 超奇异椭圆曲线

定义在有限域$\mathbb{F}_q(char(\mathbb{F}_q) \neq 2,3)$上的椭圆曲线$E$都可由形如

$$y^2 = x^3 + Ax + B$$

的方程定义，其中$A, B \in \mathbb{F}_q$满足$4A^3 + 27B^2 \neq 0$. 椭圆曲线上所有坐标在$\mathbb{F}_q$ 中的点（称为$\mathbb{F}_q$-有理点）所构成的集合记为$E(\mathbb{F}_q)$, 即

$$E(\mathbb{F}_q) = \{(x, y) \in \mathbb{F}_q^2 \,|\, y^2 = x^3 + Ax + B\} \cup \{\mathcal{O}\},$$

其中$\mathcal{O}$是无穷远点. 我们有时也谈及坐标在$\mathbb{F}_q$的代数闭包$\bar{\mathbb{F}}_q$（含$\mathbb{F}_q$上所有多项式的根的最小域）的点集$E(\bar{\mathbb{F}}_q)$.

椭圆曲线上的点对于“弦切律”定义的群运算构成一个交换群，其中无穷远点$\mathcal{O}$是单位元. 由“弦切律”可以得到$P + Q$坐标与P, Q坐标之间的代数关系. 特别地，对$E(\mathbb{F}_q)$中点进行加法和求逆运算所得点的坐标依然在$\mathbb{F}_q$中，因此，$E(\mathbb{F}_q)$也是一个群.

令$P = (x_P, y_P)$和$Q = (x_Q, y_Q)$是$E(\mathbb{F}_q)$中的不等于$\mathcal{O}$的两个点, 则$P + Q = R = (x_R, y_R)$的坐标可以这样计算：

1. 若$x_P \neq x_Q$，则$x_R = \lambda^2 - x_P - x_Q, y_R = \lambda(x_P - x_R) - y_P$, 其中$\lambda = (y_P - y_Q)/(x_P - x_Q)$；
2. 若$x_P = x_Q$，且$y_P \neq y_Q$，则$R = \mathcal{O}$；
3. 若$P = Q$，且$y_P \neq 0$，则$x_R = \lambda^2 - 2x_P, y_R = \lambda(x_P - x_R) - y_P$, 其中$\lambda = (3x_P^2 + A)/2y_P$；
4. 若$P = Q$，且$y_P = 0$，则$R = \mathcal{O}$.

有理点群$E(\mathbb{F}_q)$所含点的个数称为它的阶，记作$\#E(\mathbb{F}_q)$. 定义在特征p有限域$\mathbb{F}_q$上的椭圆曲线E称为是超奇异的，如果$\#E(\mathbb{F}_q) \equiv 1 \pmod{p}$. 事实上，它对所有的整数$n \geq 1$满足$\#E(\mathbb{F}_{q^n}) \equiv 1 \pmod{p}$. 超奇异椭圆曲线还有其他等价的定义，比如$p$-挠元群$E[p] = \{P \in E(\bar{\mathbb{F}}_q) | \, pP = \mathcal{O}\}$只含$\mathcal{O}$, 自同态环$\mathrm{End}_{\bar{\mathbb{F}}_q}(E)$是非交换的等.

椭圆曲线$E : y^2 = x^3 + Ax + B$的j-不变量定义为

$$j(E) = 1728\frac{4A^3}{4A^3 + 27B^2}.$$

如果$j(E) = 0$，则曲线形式为$y^2 = x^3 + B$；如果$j(E) = 1728$，则曲线形式为$y^2 = x^3 + Ax$. 给定$j_0 \in \mathbb{F}_q(j_0 \neq 0, 1728)$, 则椭圆曲线

$$y^2 = x^3 + \frac{3j_0}{1728 - j_0}x + \frac{2j_0}{1728 - j_0}$$

的 j-不变量等于 j_0. 定义在 $\mathbb{F}_q$ 上的超奇异椭圆曲线的 j-不变量都含在 $\mathbb{F}_{p^2}$ 中(其中 $p = \mathrm{char}(\mathbb{F}_q)$), 个数大约为 $p/12$, 其中有 $O(\sqrt{p}\log p)$ 个 j-不变量在素域 $\mathbb{F}_p$ 中.

1.1.3 同源

设 E_1、E_2 为定义在 $\mathbb{F}_q$ 上的两条椭圆曲线. 如果存在态射 $\phi : E_1 \to E_2$ 映单位元 $\mathcal{O}$ 为单位元 $\mathcal{O}$, 则称 ϕ 为同源. 同源 ϕ 通常由 E_1 上的有理函数表示. 我们只考虑定义在 $\mathbb{F}_q$ 上的同源，即 ϕ 可表示为

$$\phi((x, y)) = (f(x), yg(x)),$$

其中 $f(x), g(x)$ 是 $\mathbb{F}_q$ 上的有理函数. 我们可以将 f 表示为 $\mathbb{F}_q$ 上两个互素多项式 $p(x), q(x)$ 的商 $f(x) = p(x)/q(x)$. 则同源 ϕ 的次数定义为 $\deg(\phi) = \max\{\deg(p(x)), \deg(q(x))\}$. 如果 E_1 与 E_2 间存在定义在 $\mathbb{F}_q$ 上的同源，则称 E_1 与 E_2 是 $\mathbb{F}_q$-同源的. 椭圆曲线 E_1 与 E_2 是 $\mathbb{F}_q$-同源的当且仅当 $\#E_1(\mathbb{F}_q) = \#E_2(\mathbb{F}_q)$.

给定同源映射 $\phi : E_1 \to E_2$, 我们称

$$\ker(\phi) = \{P \in E_1(\bar{\mathbb{F}}_q) | \phi(P) = \mathcal{O}\}.$$

为 ϕ 的核. 它的阶整除 $\deg(\phi)$. 当 $\#\ker(\phi) = \deg(\phi)$ 时, 我们称 ϕ 是可分同源，否则称 ϕ 是不可分同源. 将椭圆曲线 $E_1 : y^2 = x^3 + A_1x + B_1$ 的方程系数取 $p = \mathrm{char}(\mathbb{F}_q)$ 次幂可得椭圆曲线 $E_1^{(p)} : y^2 = x^3 + A_1^p x + B_1^p$, 则 Frobenious 映射

$$Frob : E_1 \to E_1^{(p)}, (x, y) \mapsto (x^p, y^p)$$

是一个次数为 p 的不可分同源. 任一同源 ϕ 都可分解成一系列 Frobenious 映射与可分同源的复合映射.

对于 $E(\bar{\mathbb{F}}_q)$ 的任意有限子群 H, 存在同构意义下唯一的椭圆曲线 E' 和可分同源 $\phi : E \to E'$ 使得 $\ker(\phi) = H$. 这里的 E' 通常记作 E/H. 同源 $\phi = (\alpha_1, \alpha_2)$ 可如下给出:

$$\begin{aligned} \alpha_1(P) &= \sum_{Q \in H} x(Q + P) - \sum_{Q \in H, Q \neq \mathcal{O}} x(Q), \\ \alpha_2(P) &= \sum_{Q \in H} y(Q + P) - \sum_{Q \in H, Q \neq \mathcal{O}} y(Q). \end{aligned}$$

由加法公式可计算 ϕ 及 E/H 的明确表达式(Vélu 公式). 如果 H 是 $\mathbb{F}_q$-有理的，即 $\{(x^q, y^q) | (x, y) \in H\} \cup \{\mathcal{O}\} = H$, 那么存在定义在 $\mathbb{F}_q$ 的椭圆曲线 E' 和 $\mathbb{F}_q$-有理的可分同源 ϕ.

例1.1. 考虑$\mathbb{F}_{11}$上的椭圆曲线$E: y^2 = x^3 + x + 3$. 设H_1是由2阶点$(3,0)$生成的子群. 对于一般点$P = (x, y) \in E$有$P + (3,0) = (\frac{3x+8}{x+8}, \frac{5y}{(x+8)^2})$，故

$$\begin{aligned} \alpha_1(P) &= \frac{3x+8}{x+8} + x - 3 = \frac{x^2+8x+6}{x+3}, \\ \alpha_2(P) &= \frac{5y}{(x+8)^2} + y - 0 = \frac{(x^2+5x+3)}{(x-3)^2} y. \end{aligned}$$

进而可以确定α_1, α_2满足$\alpha_2^2 = \alpha_1^3 + 4\alpha_1 + 9$, 即$E/H_1$由$y^2 = x^3 + 4x + 9$定义. 设$H$是由9阶元$(4,4)$生成的子群, H_2是由$3(4,4) = (1,4)$生成的3阶子群. 则$E' = E/H_2$可由$y^2 = x^3 + 5x + 5$定义, $\phi_2 : E \to E'$由

$$(x, y) \mapsto (\frac{x^3+9x^2+9x+1}{(x+10)^2}, \frac{x^3+8x^2+6x}{(x+10)^3} y)$$

给出. 可以验证$\phi_2(4,4) = (4,10)$是E'上的3阶元. 同样可计算$E'' = E'/\langle(4,10)\rangle$的方程$y^2 = x^3 + 3x + 1$和同源$\phi_3 : E' \to E''$的表达式

$$(x, y) \mapsto (\frac{x^3+3x^2+x+9}{(x+7)^2}, \frac{x^3+10x^2+8x}{(x+7)^3} y).$$

可以验证同源$\phi_3 \circ \phi_2 : E \to E' \to E''$是以$H$为核的同源.

由于Vélu 公式的复杂度为$O(\#H)$, 对于一些特殊的情形可以通过计算同源链来降低计算复杂度(如上例). 设l是与$\mathrm{char}(\mathbb{F}_q)$互素的素数, H是由l^e阶点P生成的循环群. 则$\phi : E \to E/H$可由e个l-次同源复合得到:

- 因$l^{e-1}P$是l阶点, 故有l-次同源$\phi_0 : E \to E_1 = E/\langle l^{e-1}P\rangle$，$P$的像点$P_1 = \phi_0(P)$阶为$l^{e-1}$.
- 若已计算出$\psi : E_1 \to E_1/\langle P_1\rangle$, 则$\mathrm{Ker}(\psi \circ \phi_0) = \phi_0^{-1}(\langle P_1\rangle) = H$, 即$\phi = \psi \circ \phi_0$. 同理, 可计算$l$-次同源$\phi_1 : E_1 \to E_2 = E_1/\langle l^{e-2}P_1\rangle$及$l^{e-2}$阶点$P_2 = \phi_1(P_1)$. 我们还需计算$\varphi : E_2 \to E_2/\langle P_2\rangle$, 从而有$\psi = \varphi \circ \phi_1$, $\phi = \varphi \circ \phi_1 \circ \phi_0$. 由此可看出，一直做下去会得到$l$-次同源$\phi_0, \phi_1, \ldots, \phi_{e-1}$使得$\phi = \phi_{e-1} \circ \cdots \circ \phi_1 \circ \phi_0$.

1.1.4 起始曲线

本算法选择的起始曲线为SIDH的超奇异起始曲线

$$E_0(\mathbb{F}_{p^2}) : y^2 = x^3 + x,$$

且满足$\#E_0(\mathbb{F}_{p^2}) = (2^{e_1}3^{e_2})^2$. 其$j$-不变量为1728.

选择$E_0[2^{e_1}]$的一组$\mathbb{Z}$-基$\{P_1, Q_1\}$, $E_0[3^{e_2}]$的一组$\mathbb{Z}$-基$\{P_2, Q_2\}$, 并令$R_1 = P_1 - Q_1$, $R_2 = P_2 - Q_2$. 对于$i = 1, 2$, 记$P_i = (x_{P_i}, y_{P_i})$, $Q_i = (x_{Q_i}, y_{Q_i})$, $R_i = (x_{R_i}, y_{R_i})$.

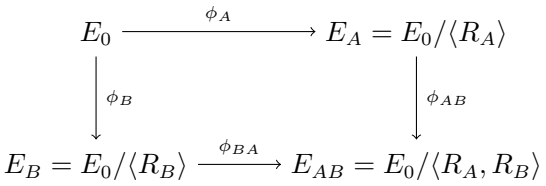

图 1: SIDH

1.2 Jao-De Feo SIDH 密钥交换以及简化描述

我们这里描述 Jao 和 De Feo 给出的基于超奇异同源的密钥交换协议-SIDH [JD14]. 令 $p = l_1^{e_1} l_2^{e_2} \cdot f \pm 1$ 为一个大素数, 其中 l_1 和 l_2 为两个小素数, 并且 f 为一个整系数. 我们可以定义一个 $\mathbb{F}_{p^2}$ 上的超奇异曲线 E_0，满足 $|E_0(\mathbb{F}_{p^2})| = (l_1^{e_1} l_2^{e_2} \cdot f)^2$. 令 $\mathbb{Z}_m$ 为模m的剩余类整环. 对于 $m \in \{l_1^{e_1}, l_2^{e_2}\}$，$E_0[m] \simeq \mathbb{Z}_m \times \mathbb{Z}_m$. 令点 P_1, Q_1 为 $E_0[l_1^{e_1}]$的生成元，点 P_2, Q_2 为 $E_0[l_2^{e_2}]$的生成元. 公共参数为$(E_0; P_1, Q_1; P_2, Q_2; l_1, l_2, e_1, e_2)$.

SIDH 密钥交换协议如图 1 所示. Alice 从 $\mathbb{Z}_{l_1^{e_1}}$ 中选择私钥 k_a 并计算同源 $\phi_A : E_0 \to E_A$，使得核为子群 $\langle R_A \rangle = \langle P_1 + [k_a]Q_1 \rangle$. 之后发送 E_A 和两个辅助点 $\phi_A(P_2), \phi_A(Q_2)$ 给 Bob. 类似地，Bob 从 $\mathbb{Z}_{l_2^{e_2}}$ 中选择 k_b 并计算同源 $\phi_B : E_0 \to E_B$ 使得核为子群 $\langle R_B \rangle = \langle P_2 + [k_b]Q_2 \rangle$. Bob 发送 E_B 和两个辅助点 $\phi_B(P_1), \phi_B(Q_1)$ 给 Alice. 为了计算会话密钥, Alice 计算同源 $\phi_{BA} : E_B \to E_{BA}$ 使得核为由 $\phi_B(P_1) + [k_a]\phi_B(Q_1)$ 生成的子群. 类似地, Bob 计算同源 $\phi_{AB} : E_A \to E_{AB}$ 使得核为由 $\phi_A(P_2) + [k_b]\phi_A(Q_2)$ 生成的子群. 由于 $\phi_{AB} \circ \phi_A$ 和 $\phi_{BA} \circ \phi_B$ 具有相同的核 $\langle R_A, R_B \rangle$, Alice 和 Bob 可以计算出相同的 j-不变量 $j(E_{AB}) = j(E_{BA})$.

由于使用 [FTTY18] 给出的简化描述，更有利于对于算法的描述. 我们这里给出简化的描述. 令 $\{t, s\} = \{1, 2\}$, 并记 $\mathfrak{g} = (E_0; P_1, Q_1, P_2, Q_2)$ 和 $\mathfrak{e} = (l_1, l_2, e_1, e_2)$. 我们定义下面的超奇异同源曲线和辅助点如下

$$\text{SSEC}_p = \{\text{supersingular elliptic curves } E \text{ over } \mathbb{F}_{p^2} \text{ with } E(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}_{l_1^{e_1} l_2^{e_2} f})^2\};$$

$$\text{SSEC}_A = \{(E; P_t', Q_t') | E \in \text{SSEC}_p, (P_t', Q_t') \text{ is basis of } E[l_t^{e_t}]\};$$

$$\text{SSEC}_B = \{(E; P_s', Q_s') | E \in \text{SSEC}_p, (P_s', Q_s') \text{ is basis of } E[l_s^{e_s}]\}.$$

令$\mathfrak{a} = k_a$ 并$\mathfrak{b} = k_b$, 我们定义,

$$\begin{aligned}
\mathfrak{g}^{\mathfrak{a}} &= (E_A; \phi_A(P_t), \phi_A(Q_t)) \in \mathrm{SSEC}_A, \\
&\qquad \text{where } R_A = P_s + [k_a]Q_s, \phi_A : E_0 \to E_A = E_0/\langle R_A \rangle; \\
\mathfrak{g}^{\mathfrak{b}} &= (E_B; \phi_B(P_s), \phi_B(Q_s)) \in \mathrm{SSEC}_B, \\
&\qquad \text{where } R_B = P_t + [k_b]Q_t, \phi_B : E_0 \to E_B = E_0/\langle R_B \rangle; \\
(\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}} &= j(E_{BA}), \text{where } R_{BA} = \phi_B(P_s) + [k_a]\phi_B(Q_s), \\
&\qquad \phi_{BA} : E_B \to E_{BA} = E_B/\langle R_{BA} \rangle; \\
(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}} &= j(E_{AB}), \text{ where } R_{AB} = \phi_A(P_t) + [k_b]\phi_A(Q_t), \\
&\qquad \phi_{AB} : E_A \to E_{AB} = E_A/\langle R_{AB} \rangle.
\end{aligned}$$

基于这些符号 SIDH 密钥交换协议更像经典的 Diffie-Hellman 密钥交换协议. 也就是公共参数为 $\mathfrak{g}$ 和 $\mathfrak{e}$. Alice 选择私钥 $\mathfrak{a}$ 并发送 $\mathfrak{g}^{\mathfrak{a}}$ 给 Bob, Bob 选择私钥$\mathfrak{b}$ 并发送 $\mathfrak{g}^{\mathfrak{b}}$ 给 Alice. 共同会话密钥为 $j = (\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}} = (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$.

1.3 具体如何表示参数以及进行同源的计算

1.3.1 如何表示曲线及辅助点

如第 1.2 所示，在密钥交换中，Alice 要发送 E_A 和两个辅助点 $\phi_A(P_2), \phi_A(Q_2)$ 给 Bob，Bob 要发送E_B 和两个辅助点$\phi_B(P_1), \phi_B(Q_1)$ 给 Alice. 其实可以使用三个坐标来表示这些信息，并且该表示方法还有理由使用更高效的同源计算方法. 文献 [CLN16] 发现其实使用三个点的横坐标就可以表示曲线 E_A 和两个点 $\phi_A(P_2) = (x_{\phi_A(P_2)}, y_{\phi_A(P_2)})$, $\phi_A(Q_2) = (x_{\phi_A(Q_2)}, y_{\phi_A(Q_2)})$ ([CLN16] 第6节“Key Generation”). 具体来说曲线 E_A 只需要一个 Montgomery 系数 a 来确定。由于给定 $(x_{\phi_A(P_2)}, x_{\phi_A(Q_2)}, x_{\phi_A(R_2)})$ (其中 $R_2 = Q_2 - P_2$, $x_{\phi_A(R_2)}$ 为 $\phi_A(R_2)$ 的横坐标) 就可以确定 Montgomery 系数 a 从而确定 E_A (具体公式见 [CLN16] 第6节“Key Generation”). 因此使用曲线 E_A 和两个点$\phi_A(P_2), \phi_A(Q_2)$ 的表示方法可以和 $(x_{\phi_A(P_2)}, x_{\phi_A(Q_2)}, x_{\phi_A(Q_2-P_2)})$ 的三点横坐标方法相互转换.

当然同源计算的初始值和中间结果也可以使用三个横坐标方法表示. 比如在公共参数中的 P_1, Q_1, P_2, Q_2 都可以使用横坐标并增加两个点 $R_1 = P_1 - Q_1$, $R_2 = P_2 - Q_2$ 来表示. 具体来说，选择 $E_0[2^{e_1}]$ 的一组 $\mathbb{Z}$-基 $\{P_1, Q_1\}$, $E_0[3^{e_2}]$ 的一组 $\mathbb{Z}$-基 $\{P_2, Q_2\}$, 并令 $R_1 = P_1 - Q_1$, $R_2 = P_2 - Q_2$. 对于$i = 1, 2$, 记 $P_i = (x_{P_i}, y_{P_i})$, $Q_i = (x_{Q_i}, y_{Q_i})$, $R_i = (x_{R_i}, y_{R_i})$.

1.3.2 如何进行同源的计算

本节讨论给定 $\mathfrak{g}$, $\mathfrak{a}$ 如何计算 $\mathfrak{g}^{\mathfrak{a}}$, 以及给定 $\mathfrak{g}^{\mathfrak{a}}$, $\mathfrak{b}$ 如何计算 $(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$ 的问题.

如 1.1.3 所示，同源的计算可以通过同源链降低计算复杂度. 也就是 l^{e_l} 次同源 ϕ 可以分解成 e_l 个 l 次同源的复合映射, 我们可以通过计算 e_l 个 l 次同源得到 ϕ. 如前所述，对每个同源 $\psi : E \mapsto E', R \mapsto \psi(R)$ 都存在有理函数 f 使得 $x_{\psi(R)} = f(x_R)$. 在下面两个计算 $\mathfrak{g}^{\mathfrak{a}}$ 和 $(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$ 的算法中，第 i 个 l 次同源 x-坐标映射所对应的函数 f 记为 f_i.

计算 $\mathfrak{g}^{\mathfrak{a}}$.

给定公共参数和私钥 $\mathfrak{g}$ 和$\mathfrak{a} \in \mathbb{Z}_{l_t^{e_t}}$ 则计算 $\mathfrak{g}^{\mathfrak{a}}$ 也就是同源 $\phi : E_0 \to E_0/\langle P_t + [\mathfrak{a}]Q_t\rangle$ 以及辅助点 $\phi(P_s), Q_s$ (其中 $\{t, s\} = \{1, 2\}$). 使用横坐标的表示方法，算法的输入为 $\mathfrak{a} \in \mathbb{Z}_{l_t^{e_t}}$,以及 $(x_{P_s}, x_{Q_s}, x_{R_s})$, $(x_{P_t}, x_{Q_t}, x_{R_t})$, 则 $\mathfrak{g}^{\mathfrak{a}}$ 如算法 1 所示.

Algorithm 1 $\mathfrak{g}^{\mathfrak{a}}$

公共参数: $p = 2^{e_1}3^{e_2} - 1$, $\mathfrak{g}$, 其中包含 E_0, $\{x_{P_1}, x_{Q_1}, x_{R_1}\}$, $\{x_{P_2}, x_{Q_2}, x_{R_2}\}$

输入: $\mathfrak{a}$

输出: $\mathfrak{g}^{\mathfrak{a}}$

令$x_S \leftarrow x_{P_t+[\mathfrak{a}]Q_t}$;

令$(x_1, x_2, x_3) \leftarrow (x_{P_s}, x_{Q_s}, x_{R_s})$;

For i from 0 to $e_t - 1$

1. 计算 l_t-同源 $\phi_i : E_i \to E'$, $(x, -) \mapsto (f_i(x), -)$,
   使得 $ker(\phi_i) = \langle [l_t^{e_t-i-1}]S\rangle$,其中 S 是 E_i 上 x-坐标为 x_S 的点.
2. 令$E_{i+1} \leftarrow E'$
3. 令$x_S \leftarrow f_i(x_S)$
4. 令$(x_1, x_2, x_3) \leftarrow (f_i(x_1), f_i(x_2), f_i(x_3))$

输出 $\mathfrak{g}^{\mathfrak{a}} = (x_1, x_2, x_3)$.

下面我们逐一解释算法 1 中的语句. 首先回忆第 1.1.3 节中“ϕ 可由两个 $\mathbb{F}_q$ 上的有理函数 $f(x)$、$g(x)$ 表示，它们满足 $\phi((x, y)) = (f(x), y \cdot g(x))$”. 并且其实只要计算横坐标$f(x)$ 就行. 其中，“令 $x_S \leftarrow x_{P_t+[\mathfrak{a}]Q_t}$ ” 就是计算 $P_t + [\mathfrak{a}]Q_t$ 的横坐标，从而确定 $R_0 = \langle P_t + [\mathfrak{a}]Q_t\rangle$. For 循环中第 1 行，就是计算 ϕ_i 并得到 E_{i+1} 使得核为 $\langle l_t^{e_t-i+1}]R_i\rangle$. 像上面回忆的一样，$\phi_i$ 可以由一个有理函数$f_i(x)$ 表示. 所以这里就是$\phi_i : E_i \to E'$可以表示成$(x, -) \mapsto (f_i(x), -)$. 那么这里的核就是 $R_i = f_{i-1}(R_{i-1})$. 这就是为什么有第 3 行 (比如当 $i = 1$ 时， $R_1 = S = (f_0(x), -)$)的原因. For 循环中第 2 行，其实就是上面计算出来的 E' 不断地变化的，也就是

$$E_0 \xrightarrow[\phi_0]{\text{or} f_0} E_1 \xrightarrow[\phi_1]{\text{or} f_1} \cdots \xrightarrow[\phi_{e_1}]{\text{or} f_{e_t}} E_{e_t}.$$

For 循环中第4行就是计算$(\phi_i((x_1,y_1)),\phi_i((x_2,y_2)),\phi_i((x_3,y_3)))$.

计算 $(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$.

给定私钥$\mathfrak{b}\in\mathbb{Z}_{l_s^{e_s}}$和$\mathfrak{g}^{\mathfrak{a}}$(具体表示为$(x_{P'_s},x_{Q'_s},x_{R'_s})$), $(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$则计算$\mathfrak{g}^{\mathfrak{a}}$对应的曲线$E'_0$, 并计算和输出曲线$E'_0/\langle P'_s+[\mathfrak{b}]Q'_s\rangle$的$j$-不变量. 如算法2所示.

Algorithm 2 $(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}$

公共参数: $p=2^{e_2}3^{e_3}-1$

输入: secret key $\mathfrak{b}$, public key $\mathfrak{g}^{\mathfrak{a}}=(x_{P'_s},x_{Q'_s},x_{R'_s})$

输出: j-不变量j

由$\mathfrak{g}^{\mathfrak{a}}$计算曲线$E'_0$; (具体方法见 [CLN16] 第6节"Key Generation")

令$x_S\leftarrow x_{P'_s+[\mathfrak{b}]Q'_s}$;

令$(x_1,x_2,x_3)\leftarrow(x_{P'_s},x_{Q'_s},x_{R'_l})$;

For i from 0 to e_s-1

1. 计算l_s-同源$\phi_i:E'_i\mapsto E'$, $(x,-)\mapsto(f_i(x),-)$,
   使得$ker(\phi_i)=\langle[l_s^{e_s-i-1}]S\rangle$, 其中$S$是$E'_i$上$x$-坐标为$x_S$的点.
2. 令$E'_{i+1}\leftarrow E'$
3. 令$x_s\leftarrow f_i(x_S)$
4. 令$(x_1,x_2,x_3)\leftarrow(f_i(x_1),f_i(x_2),f_i(x_3))$

输出$j(E'_{e_s})$.

1.4 困难问题假设

我们这里描述同源中的标准困难问题. 令$\{s,t\}=\{1,2\}$, $\mathfrak{g}$和$\mathfrak{e}$如1.2节所示.

定义1.1 (计算 SIDH 假设/SI-CDH Assumption [JD14, FTTY18]). *计算 SIDH 问题是指, 给定公共参数$\mathfrak{g}$, $\mathfrak{e}$, 和$\mathfrak{g}^{\mathfrak{a}}$, $\mathfrak{g}^{\mathfrak{b}}$, 其中$\mathfrak{a}\leftarrow\mathbb{Z}_{l_s^{e_s}}$, $\mathfrak{b}\leftarrow\mathbb{Z}_{l_t^{e_t}}$, 计算$j$-不变量$(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}=(\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}}$. 对于任意的多项式时间$\mathcal{A}$, 我们定义解决计算 SIDH 问题的优势为*

$$Adv_{\mathcal{A}}^{sicdh}=Pr[j'=(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}|j'\leftarrow\mathcal{A}(\mathfrak{g},\mathfrak{e},\mathfrak{g}^{\mathfrak{a}},\mathfrak{g}^{\mathfrak{b}})].$$

计算 SIDH 假设指: 对于任意的多项式时间的算法$\mathcal{A}$, 解决计算 SIDH 问题的优势是可忽略的.

定义1.2 (判定 SIDH 假设/ SI-DDH Assumption [JD14, FTTY18]). *令D_0和D_1为如下两个分布:*

$$D_1:=\{\mathfrak{e},\mathfrak{g},\mathfrak{g}^{\mathfrak{a}},\mathfrak{g}^{\mathfrak{b}},(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}|\mathfrak{a}\leftarrow\mathbb{Z}_{l_s^{e_s}},\mathfrak{b}\leftarrow\mathbb{Z}_{l_t^{e_t}}\}$$
$$D_0:=\{\mathfrak{e},\mathfrak{g},\mathfrak{g}^{\mathfrak{a}},\mathfrak{g}^{\mathfrak{b}},(\mathfrak{g}^{\mathfrak{s}})^{\mathfrak{t}}|\mathfrak{a},\mathfrak{s}\leftarrow\mathbb{Z}_{l_s^{e_s}},\mathfrak{b},\mathfrak{t}\leftarrow\mathbb{Z}_{l_t^{e_t}}\}$$

判定 *SIDH* 问题指对于随机选择的 $b \leftarrow \{0,1\}$ 和分布 D_b 猜测 b的问题. 对于任意的概率多项式时间 $\mathcal{A}$，解决判定 *SIDH* 问题的优势如下，

$$Adv_{\mathcal{A}}^{siddh} = Pr[b' = b | b' \leftarrow \mathcal{A}(\mathfrak{d}_b \leftarrow D_b), b \leftarrow \{0,1\}] - 1/2.$$

判定 *SIDH* 假设指: 对于任意的概率多项式时间 $\mathcal{A}$ 解决判定 *SIDH* 问题的优势是可忽略的.

1.5 认证密钥交换协议安全模型

认证密钥交换协议的模型包括BR 模型[BR93], CK 模型[CK01]，eCK 模型[LLM07]和CK$^+$模型[FSXY12]. CK$^+$ 模型是[FSXY12] 对于HMQV [Kra05] 方案安全性的总结与梳理，被认为是认证密钥交换协议最强的安全模型之一. CK$^+$ 安全模型不仅包括CK模型的基本要求，还考虑了密钥泄露伪装攻击(KCI), 最大泄漏攻击(MEX) 和弱前向安全性(wPFS)，并且支持任意注册. 因此，CK$^+$ 模型被认为是目前理论上对攻击类型覆盖最全面的安全模型.

我们本小节给出CK$^+$ 安全模型的描述[FSXY12]. 我们这里给出2轮协议的版本.

在认证密钥交换协议中, U_i 记录一个下标为i 的用户, 并且每一个用户是一个概率多项式时间的图灵机. 假设每一个用户U_i 有一对长期的公私钥对(sk_i, pk_i), 并通过CA和用户的身份U_i 进行公开可验证的绑定. 对于CA没有其他的要求，特别是不要求CA验证注册者证明对私钥信息的知晓性，或者长期公钥的合法性.

会话. 每个用户都可能被动激活一个会话- *session*. 当用户接收到$(\Pi, \mathcal{I}, U_A, U_B)$ 的消息会作为发起者发起一次会话，也可能收到消息$(\Pi, \mathcal{R}, U_B, U_A, X_A)$ 并作为应答者回复消息，其中Π 标识协议的, $\mathcal{I}$ 和$\mathcal{R}$ 分别代表发起者和应答者. 如果被激活发起一次会话$(\Pi, \mathcal{I}, U_A, U_B)$, U_A 称作会话的发起者. 被消息$(\Pi, \mathcal{R}, U_B, U_A, X_A)$激活的话, U_B 称作应答者.

根据不同协议的设定, 用户生成称作临时私钥-*ephemeral secret key* 的随机数, 计算并维持一个会话内部状态*session state*, 生成并发送消息, 计算会话密钥并擦除会话状态. Canetti-Krawczyk [CK01] 定义了会话内部状态但是没有指定具体的值. LaMacchia等[LLM07] 解释为随机数并称作临时私钥*ephemeral secret key*. 我们这里要求会话内部状态至少包含临时私钥.

一次会话也有可能在生成会话密钥前中止. 发起者U_A 生成并发送消息X_A, 之后有可能从应答者U_B 收到类型为$(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ 的消息, 并有可能计算会话密钥SK. 相反的应答者U_B 输出发送X_B, 并有可能计算会话密钥SK. 如果一次会话的拥有者计算出会话密钥，我们就称该会话完成.

每次会话有拥有者，配对者和会话id. 如果U_A 是发起者, 则会话id为$\mathsf{sid} = (\Pi, \mathcal{I}, U_A, U_B, X_A)$或者$\mathsf{sid} = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$. 该记法是指$U_A$ 是拥有者，U_B

是配对者. 如果U_B 是应答者，会话id记为$\mathsf{sid} = (\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$, 是指$U_B$ 是拥有者，U_A 是配对者. 而$(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ 的配对会话指$(\Pi, \mathcal{R}, U_B, U_A, X_A, X_B)$，反之亦然.

敌手能力. 我们通过以下方式询问的方式模拟敌手$\mathcal{A}$在真实环境中的攻击. 敌手可以控制网络并可以获取一些秘密信息.

- Send: $\mathcal{A}$ 发送下面类型的消息$(\Pi, \mathcal{I}, U_A, U_B)$, $(\Pi, \mathcal{R}, U_B, U_A, X_A)$, 或$(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$, 并获取回应.
- SessionKeyReveal(sid): 如果一个会话sid 已经完成, 则$\mathcal{A}$ 可以获取sid 的会话密钥.
- SessionStateReveal(sid): 如果会话没有完成，敌手$\mathcal{A}$ 可以获得会话sid 的内部状态. 会话状态包括所有的内部状态，但是不包括及时擦除的信息，也不包括长期私钥.
- Corrupt(U_i): 敌手$\mathcal{A}$ 可以获取U_i 的所有信息. 从U_i 被Corrupt之时起，其所有行为可以被$\mathcal{A}$所控制.

新鲜会话. 令$\mathsf{sid}^* = (\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ 或$(\Pi, \mathcal{I}, U_A, U_B, X_A, X_B)$ 为用户U_A 与用户U_B 之间的完成会话. 如果sid^* 的配对会话存在，记做$\overline{\mathsf{sid}}^*$. 我们称会话$\mathsf{sid}^*$ 为新鲜的，如果$\mathcal{A}$ 不进行如下访问: 1)如果$\overline{\mathsf{sid}}^*$ 存在SessionStateReveal(sid^*), SessionKeyReveal(sid^*), 和SessionStateReveal($\overline{\mathsf{sid}}^*$), SessionKeyReveal($\overline{\mathsf{sid}}^*$); 2)如果$\overline{\mathsf{sid}}^*$ 不存在，SessionStateReveal(sid^*) 和SessionKeyReveal(sid^*).

安全实验. 敌手$\mathcal{A}$ 可以进行如上访问，并选择一个目标会话进行攻击. 挑战者选择一个随机比特b. 如果$b = 1$, 生成并返回随机值作为会话密钥; 如果$b = 0$, 挑战者返回真实会话密钥. 敌手继续进行访问训练. 敌手猜测一个比特b', 如果$b' = b$ 则称敌手赢得实验. 敌手$\mathcal{A}$ 的优势定义为$\mathsf{Adv}_{\Pi}^{\mathrm{CK}^+}(\mathcal{A}) = \Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}$.

定义1.3. 如果以下条件满足，我们称认证密钥交换协议Π 是在CK^+ 模型下安全的.

正确性-*Correctness*: 诚实用户可以正确计算会话密钥，发生错误的概率为可忽略的.

安全性-*Soundness*: 如果以下情形之一下，对于任意多项式时间的敌手$\mathcal{A}$, 针对目标会话sid^*的优势$\mathsf{Adv}_{\Pi}^{\mathrm{CK}+}(\mathcal{A})$ 是可忽略的,

1. $\overline{\mathsf{sid}}^*$ 不存在，并且sid^* 拥有者的长期私钥泄漏给敌手$\mathcal{A}$.
2. $\overline{\mathsf{sid}}^*$ 不存在，并且sid^* 拥有者的临时私钥泄漏给敌手$\mathcal{A}$.
3. $\overline{\mathsf{sid}}^*$ 存在，并且sid^* 拥有者的长期私钥和$\overline{\mathsf{sid}}^*$ 拥有者的临时私钥泄漏给敌手$\mathcal{A}$.
4. $\overline{\mathsf{sid}}^*$ 存在，并且sid^* 拥有者的临时私钥和$\overline{\mathsf{sid}}^*$ 拥有者的长期私钥泄漏给敌手$\mathcal{A}$.
5. $\overline{\mathsf{sid}}^*$ 存在，并且sid^* 拥有者的长期私钥和$\overline{\mathsf{sid}}^*$ 拥有者的长期私钥泄漏给敌手$\mathcal{A}$.

6. $\overline{\mathsf{sid}}^*$ *存在，并且*sid^* *拥有者的临时私钥和*$\overline{\mathsf{sid}}^*$ *拥有者的临时私钥泄漏给敌手*$\mathcal{A}$*.*

如表 1上所示，CK$^+$ 安全模型包括了所有定义 1.3中的非平凡的泄漏, 其中包括了弱前向安全性 wPFS，KCI攻击, 和最大暴露 MEX攻击: 情形 1 对应 KCI 攻击. 情形 4 对应弱前向安全性 wPFS. 情形 2 和 3 对应最大暴露攻击 MEX.

Event	Case	sid^*	$\overline{\mathsf{sid}}^*$	sk_A	ek_A	ek_B	sk_B	Security
E_1	1	A	No	$\surd$	$\times$	-	$\times$	KCI
E_2	2	A	No	$\times$	$\surd$	-	$\times$	MEX
E_3	2	B	No	$\times$	-	$\surd$	$\times$	MEX
E_4	1	B	No	$\times$	-	$\times$	$\surd$	KCI
E_5	5	A or B	Yes	$\surd$	$\times$	$\times$	$\surd$	wPFS
E_6	4	A or B	Yes	$\times$	$\surd$	$\surd$	$\times$	MEX
$E_{7\text{-}1}$	3	A	Yes	$\surd$	$\times$	$\surd$	$\times$	KCI
$E_{7\text{-}2}$	3	B	Yes	$\times$	$\surd$	$\times$	$\surd$	KCI
$E_{8\text{-}1}$	6	A	Yes	$\times$	$\surd$	$\times$	$\surd$	KCI
$E_{8\text{-}2}$	6	B	Yes	$\surd$	$\times$	$\surd$	$\times$	KCI

表 1: CK$^+$ 模型敌手的能力. 如果存在的话 $\overline{\mathsf{sid}}^*$ 为 sid^* 的匹配会话. “Yes” 指 $\overline{\mathsf{sid}}^*$ 存在，“No” 则相反. sk_A (resp. sk_B) 指 A (resp. B) 的长期私钥. ek_A (resp. ek_B) 指A (resp. B) 的临时私钥. “$\surd$” 指可能泄漏给敌手, “$\times$” 指该值为安全的. “-” 意指不存在.

1.6 双密钥加密算法-2-Key PKE

我们所给出的认证密钥交换协议的基础性工具为双密钥加密算法-2-Key PKE [XLL+18, XAY+19] 的安全性定义与可分步计算的定义. 我们这里给出严格的定义.

双密钥的加密方案2-Key PKE 由四个算法构成2PKE=(KGen1, KGen0, Enc, Dec). 其中消息空间为$\mathcal{M}$ 密文空间为$\mathcal{C}$. 密钥生成算法KeyGen1 输出第一对公私钥(pk_1, sk_1), KeyGen0 输出第二对公私钥(pk_0, sk_0), 加密算法$\mathsf{Enc}(pk_1, pk_0, m)$ 输出密文$C \in \mathcal{C}$, $\mathsf{Dec}(sk_1, sk_0, C)$ 输出明文m.

- KeyGen1: 对于输入安全参数，计算并输出(pk_1, sk_1).
- KeyGen0: 对于输入安全参数，计算并输出(pk_0, sk_0).
- $\mathsf{Enc}(pk_1, pk_0, m; r)$:. 对于两个输入的公钥pk_0, pk_1 和消息$m \in \mathcal{M}$, 输出密文$C \in \mathcal{C}$. 如果该算法可以分成两步计算$\widetilde{C} = \widetilde{\mathsf{Enc}}(pk_1, m; r)$ 和$C = \overline{\mathsf{Enc}}(pk_0, \widetilde{C}, m; r)$. 我们称为可分步的.

- $\mathsf{Dec}(sk_1, sk_0, C)$: 对于输入的两个私钥sk_0, sk_1 和密文$C \in \mathcal{C}$, 输出明文m. 同样的如果解密算法是可分为两步：$\widetilde{C} = \overline{\mathsf{Dec}}(sk_0, C)$ 和$m = \widetilde{\mathsf{Dec}}(sk_1, \widetilde{C})$，我们称解密算法那是可分步的.

正确性. 对于$(pk_0, sk_0) \leftarrow \mathsf{KeyGen0}(\lambda)$, $(pk_1, sk_1) \leftarrow \mathsf{KeyGen1}(\lambda)$ 和$C \leftarrow \mathsf{Enc}(pk_0, pk_1, m; r)$, 以下成立$\mathsf{Dec}(sk_0, sk_1, C) = m$. 另外如果算法是可分步的$\widetilde{C} \leftarrow \widetilde{\mathsf{Enc}}(pk_1, m; r)$, $C \leftarrow \overline{\mathsf{Enc}}(pk_0, \widetilde{C}, m; r)$, 我们有$\widetilde{C} = \overline{\mathsf{Dec}}(sk_0, C)$.
安全性. 我们同时定义方案的[OW-CPA, OW-CPA] 安全性.

Game $[\mathsf{OW\text{-}CPA}, \cdot]$	Game $[\cdot, \mathsf{OW\text{-}CPA}]$
$(pk_1, sk_1) \leftarrow \mathsf{KGen1}(pp)$	$(pk_0, sk_0) \leftarrow \mathsf{KGen0}(pp)$
$(st; pk_0^*) \leftarrow \mathcal{A}_1(pk_1)$	$(st; pk_1^*) \leftarrow \mathcal{A}_1(pk_0)$
$m \leftarrow \mathcal{M}$	$m \leftarrow \mathcal{M}$
$c^* \leftarrow \mathsf{Enc}(pk_1, pk_0^*, m)$	$c^* \leftarrow \mathsf{Enc}(pk_1^*, pk_0, m)$
$m' \leftarrow \mathcal{A}_2(st, c^*)$	$m' \leftarrow \mathcal{A}_2(st, c^*)$
return $m' \stackrel{?}{=} m$	return $m' \stackrel{?}{=} m$

图 2: The games for 2-key PKE.

令$\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ 为抵抗2PKE 中pk_1的敌手. 我们定义其在游戏[OW-CPA, $\cdot$] 的优势为: $\mathrm{Adv}_{\mathsf{2PKE}}^{[\mathsf{OW\text{-}CPA}, \cdot]}(\mathcal{A}) = \Pr[\mathsf{OW\text{-}CPA}^{\mathcal{A}} \Rightarrow 1]$, 其中游戏[OW-CPA, $\cdot$] 在图2中定义.

对于任意多项式的敌手，如果$\mathrm{Adv}_{\mathsf{2PKE}}^{[\mathsf{OW\text{-}CPA}, \cdot]}(\mathcal{A})$是可忽略的，我们称2PKE 为[OW-CPA, $\cdot$] 安全的. 同样的我们可以定义[$\cdot$, OW-CPA] 安全性. 我们称方案为[OW-CPA, OW-CPA] 安全的，如果方案既是[OW-CPA, $\cdot$] 安全又是[OW-CPA, OW-CPA] 安全的.

2 算法描述

SIAKE 算法的基本组件为基于同源的双密钥加密算法 (2-key PKE)，我们首先描述基于同源的 2-key PKE (第 2.1 节)，其次给出 SIAKE (第 2.2 节).

2.1 基于同源计算的 2-Key PKE

如第 1.2 节，选择 $p = l_1^{e_1} l_2^{e_2} \cdot f \pm 1, E_0, \{P_1, Q_1\}, \{P_2, Q_2\}$. 令 $H : \{0,1\}^* \to \{0,1\}^n$ 为从pair-wise independent 哈希 [ML81]函数族 $\mathcal{H}$ 中随机选取的哈希函数. 令 $\mathfrak{g} = (E_0; P_1, Q_1, P_2, Q_2)$, $\mathfrak{e} = (l_1, l_2, e_1, e_2)$，以及 h 为公共参数 pp. 这里的 $\{s,t\} = \{1,2\}$.

特别地，本算法选择的起始曲线为

$$E_0(\mathbb{F}_{p^2}) : y^2 = x^3 + x.$$

且 $\#E_0(\mathbb{F}_{p^2}) = (l_1^{e_1} l_2^{e_2})^2$, 其 j-不变量为 1728.

选择 $E_0[l_1^{e_1}]$ 的一组 $\mathbb{Z}$-基 $\{P_1, Q_1\}$, $E_0[l_2^{e_2}]$ 的一组 $\mathbb{Z}$-基 $\{P_2, Q_2\}$, 并令 $R_1 = P_1 - Q_1$, $R_2 = P_2 - Q_2$. 对于$i = 1, 2$, 记 $P_i = (x_{P_i}, y_{P_i})$, $Q_i = (x_{Q_i}, y_{Q_i})$, $R_i = (x_{R_i}, y_{R_i})$.

基于第 1.2 节以及上述给出的基本 SIDH 计算和符号，我们这里给出基于同源计算的 2-Key PKE $2\mathsf{PKE}_{sidh}$. 说明其加密和解密算法为可分步计算的，并证明其基于 SI-DDH 假设是 [OW-CPA, OW-CPA] 安全的. 其中明文空间为 $\mathcal{M} = \{0,1\}^n \times \{0,1\}^n$, 随机数空间为 $\mathcal{R} = \mathbb{Z}_{l_t^{e_t}}$. 算法描述如下以及图 3 所示.

- KGen1(n,pp): 输入为安全参数 n 和公共参数 pp. 随机选择 $\mathfrak{a}_1 \leftarrow \mathbb{Z}_{l_s^{e_s}}$ 并计算$\mathfrak{g}^{\mathfrak{a}_1}$. 最终输出私钥和公钥
$$sk_1 := \mathfrak{a}_1, pk_1 := \mathfrak{g}^{\mathfrak{a}_1}.$$

- KGen0(n,pp): 输入为安全参数 n 和公共参数 pp. 随机选择 $\mathfrak{a}_0 \leftarrow \mathbb{Z}_{l_s^{e_s}}$ 并计算$\mathfrak{g}^{\mathfrak{a}_0}$. 最终输出私钥和公钥
$$sk_0 := \mathfrak{a}_0, pk_0 := \mathfrak{g}^{\mathfrak{a}_0}.$$

- Enc(pk_1, pk_0, m): 输入为公钥 pk_1, pk_0 和消息 $m = m_1 || m_0 \in \{0,1\}^{2n}$. 随机选择$\mathfrak{b} \leftarrow \mathbb{Z}_{l_t^{e_t}}$ 并计算$\mathfrak{g}^{\mathfrak{b}}$，$h((\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}) \oplus m_1$ 以及$h((\mathfrak{g}^{\mathfrak{a}_0})^{\mathfrak{b}}) \oplus m_0$. 加密密文为
$$c := \left(\mathfrak{g}^{\mathfrak{b}}, H\left((\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}\right) \oplus m_1, H\left((\mathfrak{g}^{\mathfrak{a}_0})^{\mathfrak{b}}\right) \oplus m_0\right).$$

- Dec(sk_1, sk_0, c): 输入为私钥$sk_1 = \mathfrak{a}_1$, $sk_0 = \mathfrak{a}_0$ 和密文$c = (\mathfrak{c}_1, c_2, c_3)$. 计算$m_1 := c_2 \oplus H\left(\mathfrak{c}_1^{\mathfrak{a}_1}\right)$ 和$m_0 := c_3 \oplus H\left(\mathfrak{c}_1^{\mathfrak{a}_0}\right)$. 明文为$m = m_1 || m_0$.

$2\mathsf{PKE}_{sidh}$ 的正确性容易验证.

注1: 通过将 pk_0 和 sk_0 设定为空值，并令密文为 $\mathfrak{c}_1, c_2$, 上述方案为基于同源的 ElGamal 方案并且在 SI-DDH 假设下为 OW-CPA 安全的 (实际上也是 IND-CPA 安全的). 也就是当 $pk_0 = 0$，我们定义 $\mathsf{Enc}(pk_1, pk_0, m)$ 输出为 $c := \left(\mathfrak{g}^{\mathfrak{b}}, H\left((\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}\right) \oplus m_1\right)$，相应地，当 $sk_0 = 0$，我们定义 $\mathsf{Dec}(sk_1, sk_0, m)$ 也只处理 ElGamal 部分.

图 3: 基于同源计算的 $2\mathsf{PKE}_{sidh}$

KGen1

输入: n, pp

输出: pk_1, sk_1

1 : $sk_1 := \mathfrak{a}_1 \leftarrow \mathbb{Z}_{l_s^{e_s}}$

2 : $pk_1 := \mathfrak{g}^{\mathfrak{a}_1}$.

输出 (pk_1, sk_1).

KGen0

输入: n, pp

输出: pk_0, sk_0

1 : $sk_0 := \mathfrak{a}_0 \leftarrow \mathbb{Z}_{l_s^{e_s}}$

2 : $pk_0 := \mathfrak{g}^{\mathfrak{a}_0}$.

输出 (pk_0, sk_0).

Enc

输入: $pk_1, pk_0, m_1 \times m_0 \in \mathcal{M}, \mathfrak{b} \in \mathcal{R}$

输出: c

1 : $\mathfrak{c}_1 \leftarrow \mathfrak{g}^{\mathfrak{b}}$

2 : $j_1 \leftarrow (\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}, j_0 \leftarrow (\mathfrak{g}^{\mathfrak{a}_0})^{\mathfrak{b}}$

3 : $c_2 \leftarrow H(j_1) \oplus m_1, c_3 \leftarrow H(j_0) \oplus m_0$.

输出： $c = (\mathfrak{c}_1, c_2, c_3)$.

Dec

输入: $sk_1 = \mathfrak{a}_1, sk_0 = \mathfrak{a}_0, c = (\mathfrak{c}_1, c_2, c_3)$

输出: m

1 : $j_1 \leftarrow \mathfrak{c}_1^{\mathfrak{a}_1}, j_0 \leftarrow \mathfrak{c}_1^{\mathfrak{a}_0}$

2 : $m_1 \leftarrow H(j_1) \oplus c_2$

3 : $m_0 \leftarrow H(j_0) \oplus c_3$.

输出： $m_1 || m_0$.

下面引理和证明同 [XXW+19] 中的 Lemma 2.

引理2.1. *在 SI-DDH假设下, $2PKE_{sidh}$ 为 [OW-CPA, OW-CPA] 安全. 严格来说, 对于任意概率多项式时间[OW-CPA, ·] (resp. [·, OW-CPA]) 敌手$\mathcal{A}$ (resp. $\mathcal{C}$), 存在算法$\mathcal{B}$ (resp. $\mathcal{D}$) 使得*

$$Adv^{[\mathsf{OW\text{-}CPA},\cdot]}_{\mathsf{2PKE_{sidh}}}(\mathcal{A}) \leq 2Adv^{siddh}_{\mathcal{B}} + 2^{-n} + 2^{-(k-n)/2},$$

$$(resp.\ \ Adv^{[\cdot,\mathsf{OW\text{-}CPA}]}_{\mathsf{2PKE_{sidh}}}(\mathcal{C}) \leq 2Adv^{siddh}_{\mathcal{D}} + 2^{-n} + 2^{-(k-n)/2}),$$

其中 $k = \log p - 4$

Proof. 我们将方案的 [OW-CPA, ·] 安全性归约到底层的SI-DDH 假设. 对于方案的 [·, OW-CPA] 安全性可简单推广得到. [OW-CPA, ·] 通过 Game 0-3 过渡证明. 我们记在 Game i 中输出1 的概率为Succ_0.

$\underline{\mathcal{B}(\mathfrak{e}, \mathfrak{g}, \mathfrak{g}_1, \mathfrak{g}_2, j)}$
01 $pk_1 \leftarrow \mathfrak{g}_1$
02 $pk_0^*, \text{state} \leftarrow \mathcal{A}(pk_1)$
03 $m_1 \leftarrow \{0,1\}^n$
04 $\mathfrak{c}_1^* = \mathfrak{g}_2$, $c_2^* = h(j) \oplus m_1$, $c_3^* \leftarrow \{0,1\}^n$
05 $c^* = (\mathfrak{c}_1^*, c_2^*, c_3^*)$
06 $m_1' || m_0' \leftarrow \mathcal{A}(\text{state}, c^*)$
07 If $m_1' = m_1$, $b' = 1$, else $b' \leftarrow \{0,1\}$.

图 4: SI-DDH 敌手$\mathcal{B}$

Game 0: 该游戏为图 2 所示的原始 [OW-CPA, ·]游戏. 则我们有

$$\text{Adv}_{\text{2PKE}}^{[\text{OW-CPA},\cdot]}(\mathcal{A}) = \text{Succ}_0.$$

Game 1: 在该游戏中我们修改[OW-CPA, ·] 挑战游戏输出 1 的条件，如果$m_1' = m_1$，则返回 1. 我们注意到在 Game 1 中只有当$m_1' = m_1$ 并且$m_0' = m_0$ 游戏才输出 1. 因此我们有$\Pr[\text{Succ}_0] \leq \Pr[\text{Succ}_1]$.

Game 2: 在该游戏中我们修改挑战密文的计算. 具体来讲, $(\mathfrak{g}^{\mathfrak{b}})^{\mathfrak{a}_1}$ 被替换为一个随机的j-不变量j^*. 如果存在一个敌手$\mathcal{A}$ 在Game 1 和Game 2 中使得输出 1 的概率有显著差距，我们构造一个算法$\mathcal{B}$ 来解决SI-DDH 问题(也就是说给定问题实例$(\mathfrak{g}, \mathfrak{g}_1, \mathfrak{g}_2, j)$). 如图 4 所示.

如 $(\mathfrak{g}, \mathfrak{g}_1, \mathfrak{g}_2, j)$ 为一个SI-DDH 实例, 则 $\mathcal{B}$ 完美模拟了 Game 1, 否则$\mathcal{B}$ 完美的模拟 Game 2. 在 SI-DDH 问题挑战中, 我们有

$$\begin{aligned}
\text{Adv}_{\mathcal{B}}^{siddh} &= \Pr[b = b'] - 1/2 \\
&= 1/2(\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]) \\
&= 1/2(\Pr[b' = 1 | \text{Game 1}] - \Pr[b' = 1 | \text{Game 2}]) \\
&= 1/2(\Pr[\text{Succ}_1] - \Pr[\text{Succ}_2]).
\end{aligned}$$

Game 3: 在该游戏中我们进一步修改挑战密文的生成. 具体来说，$H(j^*)$ 替换为一个随机比特串$h^* \leftarrow \{0,1\}^n$. 现在c_2^* 是一个完全随机串. 也就是说, 任意敌手计算出m_1 的优势为$\Pr[\text{Succ}_3] = 2^{-n}$. 注意到由于$H$ 是一个 pairwise independent 哈希函数, 由 leftover hash 引理 [BDK+11], $|\Pr[\text{Succ}_2] - \Pr[\text{Succ}_3]| \leq 2^{-(k-n)/2}$.

综上所述，我们有$\Pr[\text{Succ}_0] \leq 2\text{Adv}_{\mathcal{B}}^{siddh} + 2^{-n} + 2^{-(k-n)/2}$.

□

引理2.2. *2PKE$_{sidh}$ 双密钥加密算法是可分步计算的，并且无解密错误。*

加密算法和解密算法的分步计算如图5 所示. 无解密错误是显然的。

图 5: 2PKE$_{sidh}$中加解密算法的分步计算

$\widetilde{\mathsf{Enc}}$

输入: $pk_1 = \mathfrak{g}^{\mathfrak{a}_1}, m_1||m_0, \mathfrak{b} \in \mathcal{R}$

输出: $\tilde{c}$

1： $\tilde{\mathfrak{c}}_1 \leftarrow \mathfrak{g}^{\mathfrak{b}}$

2： $j_1 \leftarrow (\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{b}}$

3： $\tilde{c}_2 \leftarrow h(j_1) \oplus m_1, \tilde{c}_3 = m_0.$

输出：$\tilde{c} = (\tilde{\mathfrak{c}}_1, \tilde{c}_2, \tilde{c}_3).$

$\overline{\mathsf{Enc}}$

输入: $pk_0 = \mathfrak{g}^{\mathfrak{a}_0}, \tilde{c} = (\tilde{\mathfrak{c}}_1, \tilde{c}_2, \tilde{c}_3), \mathfrak{b} \in \mathcal{R}$

输出: c

1： $\mathfrak{c}_1 \leftarrow \tilde{\mathfrak{c}}_1, c_2 \leftarrow \tilde{c}_2$

2： $j_0 \leftarrow (\mathfrak{g}^{\mathfrak{a}_0})^{\mathfrak{b}}$

3： $c_3 \leftarrow h(j_0) \oplus \tilde{c}_3.$

输出：$c = (\mathfrak{c}_1, c_2, c_3).$

$\overline{\mathsf{Dec}}$

输入: $sk_0 = \mathfrak{a}_0, c = (\mathfrak{c}_1, c_2, c_3)$

输出: $\tilde{c}$

1： $j_0 \leftarrow \mathfrak{c}_1^{\mathfrak{a}_0}$

2： $\tilde{c}_3 \leftarrow h(j_0) \oplus c_3.$

输出：$\tilde{c} = (c_1, c_2, \tilde{c}_3).$

$\widetilde{\mathsf{Dec}}$

输入:$sk_{l,1}, \tilde{c} = (c_1, c_2, \tilde{c}_3)$

输出:m

1： $j_1 \leftarrow \mathfrak{c}_1^{\mathfrak{a}_1}$

2： $m_1 \leftarrow h(j_1) \oplus c_2.$

输出：$m_1||\tilde{c}_3.$

2.2 SIAKE组成

SIAKE是有四个阶段组成，其中分成用户注册阶段(SIAKE.Reg)、发起者发起阶段(SIAKE.A.int)，接收者回复计算会话密钥阶段(SIAKE.B.Shared)、发起者计算会话密钥阶段(SIAKE.A.Shared). 具体协议的执行如图 6 所示.

公共参数设定:

令 $\mathfrak{e} = (l_1, l_2, e_1, e_2)$，$\mathfrak{g} = (E_0; P_1, Q_1, P_2, Q_2)$. 令 $f : \{0,1\}^{2n} \to \mathcal{M}$, $G : \mathcal{D}_{pk_0} \times \mathcal{M} \to \mathcal{R}$, $h : \mathcal{D}_{pk_0} \times \mathcal{M} \to \{0,1\}^n$, $\hat{H} : \{0,1\}^* \to \{0,1\}^n$, $h' : \{0,1\}^n \times \mathcal{D}_{pk_0} \times \mathcal{C} \to \{0,1\}^n$ 为哈希函数. 令$H : \{0,1\}^* \to \{0,1\}^n$ 为从pair-wise independent 哈希 [ML81]函数族 $\mathcal{H}$ 中随机选取的哈希函数.

用户注册 SIAKE.Reg:

任意用户注册两套长期公私钥对. 其中一套用于用户作为发起者, 另外一套用于用户作为接收者. 具体来讲对于用户 U_A, 其首先选择 $sk_{A_1} := \mathfrak{a}_1 \in \mathbb{Z}_{l_1^{e_1}}, s_{A1}, s'_{A1} \leftarrow \{0,1\}^n)$ 并计算 $pk_{A_1} := \mathfrak{g}^{\mathfrak{a}_1}$ 作为发起者长期公私钥. 另外选择 $sk_{A_2} := (\mathfrak{a}_2 \in \mathbb{Z}_{l_2^{e_2}}, s'_{A2}, s_{A2} \leftarrow \{0,1\}^n)$ 并计算 $pk_{A_2} := \mathfrak{g}^{\mathfrak{a}_2}$ 作为接收者长期公私钥. 另外用户 U_B 也同样选择两套长期公私钥.

发起者发起 SIAKE.A.int:

- 用户 U_A 随机选择 $\mathfrak{r}_\mathfrak{o} \leftarrow \mathbb{Z}_{l_1^{e_1}}$ 作为两个随机数. 首先计算 $(pk_0, sk_0) \leftarrow \mathsf{KGen0}(\mathfrak{r}_\mathfrak{o})$, 也就是计算 $pk_0 := \mathfrak{g}^{\mathfrak{r}_\mathfrak{o}}$, $sk_0 := \mathfrak{r}_\mathfrak{o}$.
- 随机选择 $r_A \leftarrow \{0,1\}^n$, 计算 $f(s'_{A1}, r_A)$ 并令前 n 比特为 m_A, $\mathfrak{r}_A := G(0, m_A||0^n)$. 之后 U_A 计算 $C_A = \mathsf{Enc}(pk_{B2}, 0, m_A; \mathfrak{r}_A)$ 以及 $K_A = h(0, m_A)$. 也就是 $C_A = (\mathfrak{g}^{\mathfrak{r}_A}, H\left((\mathfrak{g}^{\mathfrak{b}_2})^{\mathfrak{r}_A}\right) \oplus m_A)$
- U_A 发送 C_A, pk_0 给 U_B.

回复者回复及计算会话密钥 SIAKE.B.Shared:

- 用户 U_B随机选择 $r_B \leftarrow \{0,1\}^n$, 计算 $m_B := m_{B1}||m_{B0} \leftarrow f(s'_{B2}, r_B)$, $\mathfrak{r}_B := G(pk_0, m_B)$. 之后 U_B 计算 $C_B = \mathsf{Enc}(pk_{A1}, pk_0, m_B; \mathfrak{r}_B)$ 以及 $K_B = h(pk_0, m_B)$. 也就是
$$C_B = (\mathfrak{g}^{\mathfrak{r}_B}, H\left((\mathfrak{g}^{\mathfrak{a}_1})^{\mathfrak{r}_B}\right) \oplus m_{B1}, H\left((\mathfrak{g}^{\mathfrak{r}_\mathfrak{o}})^{\mathfrak{r}_B}\right) \oplus m_{B0}).$$
- 用户 U_B 发送 C_B 给 U_A.
- 令 $(c_{A1}, c_{A2}) \leftarrow C_A$, 用户 U_B 计算 $m'_A = \mathsf{Dec}(sk_{B2}, 0, C_A)$，也就是$m'_A = H\left((\mathfrak{g}^{\mathfrak{r}_A})^{\mathfrak{b}_2}\right) \oplus c_{A2}$. 如果 $m'_A = \perp$ 或者 $C_A \neq \mathsf{Enc}(pk_{B2}, 0, m'_A; \mathfrak{r}_A)$，令 $K'_A = h'(s_{B2}, 0, C_A)$, 否则 $K'_A = h(0, m'_A)$.
- 计算会话密钥为 $SK = \hat{H}(U_A, U_B, pk_A, pk_B, C_A, pk_0, C_B, K'_A, K_B)$

发起者计算会话密钥 SIAKE.A.Shared:

- 令 $(c_{B1}, c_{B2}, C_{B3}) \leftarrow C_B$, 用户 U_A 计算 $m'_B = \mathsf{Dec}(sk_{A1}, pk_0, C_B)$，也就是$m'_{B1} = H\left((\mathfrak{g}^{\mathfrak{r}_B})^{\mathfrak{a}_1}\right) \oplus c_{B2}$, $m'_{B0} = H\left((\mathfrak{g}^{\mathfrak{r}_B})^{\mathfrak{r}_\mathfrak{o}}\right) \oplus c_{B3}$. 如果 $m'_B = m'_{B1}||m'_{B0} = \perp$ 或者 $C_B \neq \mathsf{Enc}(pk_{A1}, pk_0, m'_B; G(pk_0, m'_B))$，令 $K'_B = h'(s_{A1}, pk_0, C_B)$, 否则 $K'_B = h(pk_0, m'_B)$.
- 计算会话密钥为 $SK = \hat{H}(U_A, U_B, pk_A, pk_B, C_A, pk_0, C_B, K_A, K'_B)$

U_A的 session-state 包括 $r_A, \mathfrak{r}_\mathfrak{o}$, K'_B 以及K_A. U_B 的 session-state 包括 r_B, K_B, 但是不包括K'_A.

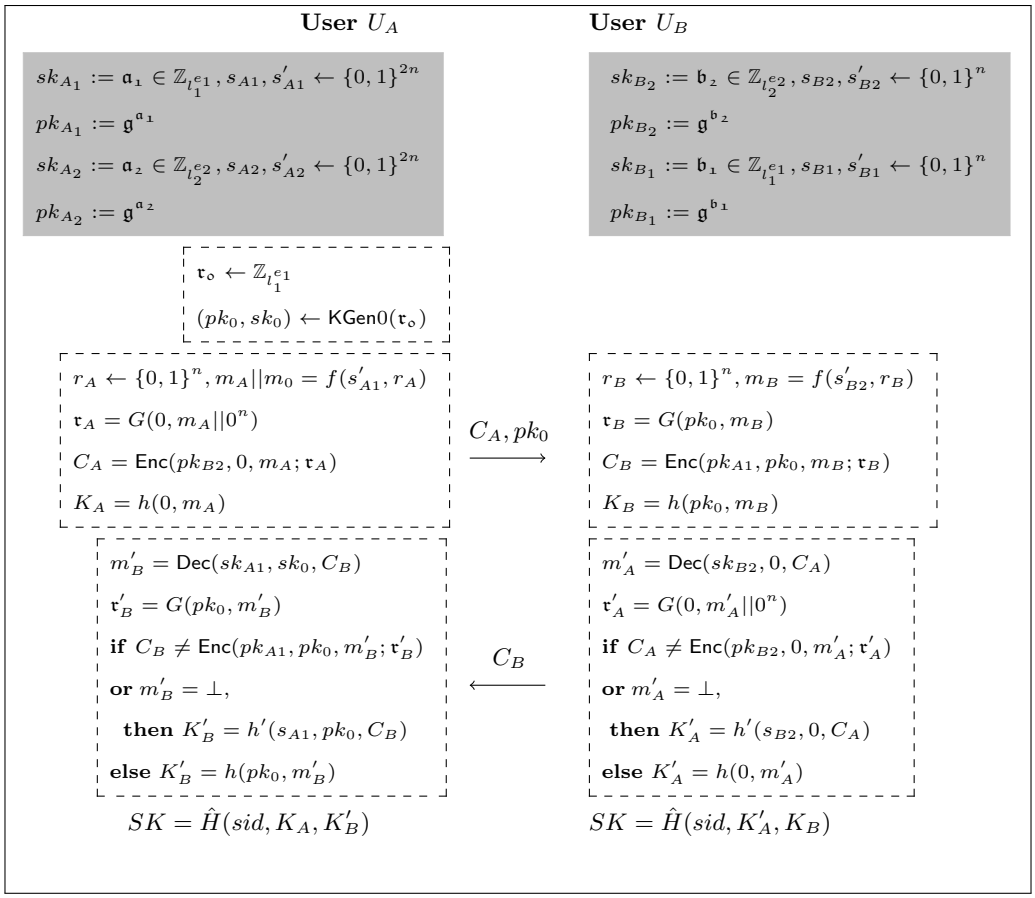

图 6: SIAKE 算法. $sid = (U_A, U_B, pk_A, pk_B, C_A, pk_0, C_B)$

3 设计原理

本节我们说明SIAKE的设计策略和主要参数选择. 在3.1 节我们给出设计的出发点与如此设计的理由，在3.2 给出具体的参数选择.

3.1 主要策略

我们设计该算法的出发点是给出抗量子攻击的通信量极低的认证密钥交换协议. 在抵抗量子攻击的几类基本方案中，超奇异同源类方案是通信量最低的，因此我们选用超奇异同源问题构造认证密钥交换协议.

3.1.1 主要问题

在经典假设下，HMQV 方案 [Kra05]是最优秀的认证密钥交换协议之一. 其不仅达

到几乎最高的安全要求，还具有通信量小和计算效率快的特点. 其安全性主要依赖于经典循环群上的 Gap 问题(计算问题与判定问题之间的 Gap问题)困难的假设. 然而在超奇异同源上的困难问题(以及格上的问题) Gap问题是简单的 [UJ18, Gal18]，无法直接借鉴经典方案 HMQV的设计技巧.

不仅如此，基于同源假设设计认证密钥交换协议还面临如下几个问题：首先，在同源结构上缺少 HMQV等经典技术中丰富的代数结构(具体指类似g^{ad+x}的计算)，无法使用；其次，和经典椭圆曲线上点的验证不同，同源计算中公钥的合法性验证是一个困难问题 [UJ18],所以敌手任意注册和任意发送非法消息的风险更大；最后，和经典椭圆曲线群不同，如果一个公钥设置为长期公钥，极易受到自适应攻击 [GPST16],从而完全泄漏长期私钥.

3.1.2 前人工作

因此，目前有两类解决办法. 一类是 Galbriath [Gal18] 将不需要丰富数学结构的经典方案 Jeong-Katz-Lee 扩展到同源问题中，或者 Fujioka 等人 [FTTY18] 将 NAXOS扩展到同源难问题中；另一类是使用 FSXY[FSXY13]的框架性结构从 CCA安全的密钥封装组合出认证密钥交换协议.

其中第一类的解决办法，无法完整解决上述问题，比如 [Gal18] 中方案不可以任意注册，无法提供 KCI安全性，前向安全性；而 [FTTY18] 无法支持任意注册，无法提供 KCI和 MEX安全性. 第二种办法虽然提供了很高的安全性，但是仍然需要较大的通信量和较多的计算量.

3.1.3 我们的思路

我们的主要思路是基于团队在 ASIACRYPT 2018 和 ASIACRYPT 2019 的工作 [XLL+18, XXW+19]以及后续工作[XAY+19]. 认证密钥交换协议需要两组长期公钥与临时公钥的特殊组合来保证方案的功能性和安全性. 在上述工作中我们发现认证密钥交换协议的基本组件是一种我们所定义的强 CCA安全的双密钥密钥封装机制，几乎所有达到强安全的认证密钥交换协议都是基于此工具构造，例如著名的 HMQV、NAXOS等方案. 具体来讲，其中一套公钥为长期公钥，另一套公钥为临时公钥.

同时我们证明通过加强版的 Fujisaki-Okamoto 转化可以从 [OW-CPA, OW-CPA]安全双密钥PKE (第2.1节) 转化为强 CCA安全的双密钥密钥封装机制从而得到强安全的认证密钥交换协议.

另一方面，在 [XXW+19]工作中我们指出，基于同源问题可以设计出和经典单密钥密文同长度的 [OW-CPA, OW-CPA]安全双密钥PKE，从而做到通信量极低的认证密钥交换协议.

然而，上述技术与方案还只能达到经典随机预言模型的安全性. 在量子随机预言模型下，加密算法的通用技术是使用确定性加密的单映射性质来避免对于私钥的依赖，从而证明方案的安全性. 但是由于我们使用 [OW-CPA, OW-CPA]安全双密钥 PKE，确定性加密的单映射性质无法全程保证. 因此基于我们在 [XAY+19] 中引入分步可计算的双密钥 PKE，并采用中间相遇单射的技术完成 SIAKE 的证明. 如同第 2.1 节以及 [XAY+19] 所述，本文的双密钥 PKE是可分步计算的，从而结合 [XAY+19] 的证明，即可得到 SIAKE 在量子随机预言模型下的安全性.

综上所述，我们采用同源问题为底层问题，并从建立[OW-CPA, OW-CPA]安全双密钥 PKE出发，通过加强版的 Fujisaki-Okamoto转化，并应用 ASIACRYPT 2018 和 ASIACRYPT 2019中的框架，完全解决上述问题，并构造出强安全的 SIAKE 方案.

3.2 参数选择以及满足要求的指定参数

我们所设计方案可以依赖任意底层的 SIDH方案，具体参数选择是和 David Jao等人提交给 NIST的无认证 SIDH方案一致的，分别为 SIAKEp503, SIAKEp751 和 SIAKE964, 其中数字为所选择素数 p的比特长度. 三套参数中都令 $l_1 = 2, l_2 = 3$. 如后面第 4.3 节所示，其中 SIAKEp751 和 SIAKE964 均能满足第二轮提交参数要求，即：提供至少量子128比特的安全性，共享密钥长度至少为256比特，无解密错误，以及满足双向认证和 CK^+ 安全模型下的安全性证明.

如第 1.1.3 所述，公共参数中的曲线和基点使用三个点的横坐标表示.

3.2.1 SIAKE503

p =
00000004 066F5418 11E1E604 5C6BDDA7 7A4D01B9 BF6C87B7 E7DAF130 85B-
DA221 1E7A0ABF 809FFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFF-
F FFFFFFFF FFFFFFFF
e2 =
000000FA
e3 =
0000009F
xQ20 =
00097453 912E12F3 DAF32EEF FD618BD9 3D3BBBF3 99137BD3 9858CADE FAE382E4
2D6E60A6 2FD62417 AD61A14B
60DB2612 5273EC98 0981325D 86E55C45 E3BB46B1
xQ21 =
00000000
yQ20 =
0009B666 40A4CC79 F82B68D7 26092338 12DF76E8 B0422EF3 527A1F2A 9915EFF1
6E094004 0DF4A15A 84A5ACF0 24FC2ED8 A50102A7 31E8D20D 033B4803 5B63DD62
yQ21 =
00000000
xP20 =
001F6D52 A7563BB9 356B98A1 16A0CA97 75DBB738 2EB29E24 E45299D8 939959EA
EEB47FF3 113F6088 2D12103E 4B8B8CD2 B97DA146 57AE8C12 8BE82209 D2DDFCA9
xP21 =
002D44C3 FAD24E4C BDDC8A2D 9DE336A9 2A9912EE 6D09E2DD 5C33AB26 D60A268A
C91F38E1 AF4C2D5B FA2B87DD 55C8CA60 19C6B0C0 8ED92B5A EB6C65A8 E06E53E9
yP20 =
003C9F7C397283C0871F78D9 F74ECC0A8F89579CCBEF8F E60D07338A F0A0322E
3F0C66CA 826AA5BF 85EB5366 6C272C8E AEC9B808 B3B78E64 22330617 AC23D6F2
yP21 =
0038222A E95DA234 ABD1B90F D897C2E2 E7995B2C 0006DC92 CC079B7C 60C94DCA
E9961CC7 A4BAEAC9 D294F6D5 760D4D65 4821193A E92AD42A C0047ADE 55C343FC
xR20 =
00173775 ECBEC79C 78FD1ED5 FE36075A ACE1F53F 8FFB97D2 A7E80DFC 2875E77E
C72D1D4A 99E13353 EC9D147B ADD96126 948A72B3 0BDD7CEB AD7B54F8 D-

DB5CD06
xR21 =
0002EAA2 24DDDA14 9BBBB908 9D2B2C47 1D068ECA 203465CE 97DBC1C8 ED0EBB0F F90E4FBE 7E266BBA 99CBAE05 1797B4D3 5D28E36C 1B1CB994 AEEED1CB 59FE5015
xQ30 =
001E7D6E BCEEC9CF C47779AF FD696A88 A971CDF3 EC61E009 DF55CAF4 B6E01903 B2CD1A12 089C2ECE 106BDF74 5894C14D 7E39B699 7F70023E 0A23B4B3 787E-F08F
xQ31 =
00000000
yQ30 =
002EC0AA EF9FBBDD 75FBDA11 DA19725F 79E842FB C355071F D631C1CD F90E08E6 01929FAE C5DAEB0D 96BBB4AD 50FC7C8A D47064F0 5C06DC5D 4AAE61CC C-EFF1F26
yQ31 =
00000000
xP30 =
0021B709 8B640A01 D88708B7 29837E87 0CFF9DF6 D4DF86D8 6A7409F4 1156CB5F 7B851482 2730940C 9B51E0D9 821B0A67 DD7ED98B 9793685F A2E22D6D 89D66A4E
xP31 =
002F37F5 75BEBBC3 3851F75B 7AB5D89F C3F07E4D F3CC5234 9804B8D1 7A17000A 42FC6C57 34B9FCFD E669730F 3E8569CE B53821D3 E8012F7F 391F5736 4F402909
yP30 =
0000078F 8A30AB36 B301BDF6 72D9E351 8AF741F8 227CC95A 9F351B99 623A826D E3F8D90D D6ED42FF 298E394E 77B7AEFE E6010CDF 34A7DE9F 9E239B10 3E7B3EEE
yP31 =
0037F3C6 00488EBB 6B11462C 4CAFC41C D5DC611A 9B0C804E 3BF50D6D 8F75C4E7 A136E29E 00D80EB8 653CA830 F2AED61D 04F9F3A8 317F7916 E016F273 3B828AC0
xR30 =
000D4818 D120A24A BF48DB51 D129E6B1 F24F4BBB 2C16FACC 0C8C0632 3EEEC2FA 5B5E887E 17226417 B1907310 BFE6784F DEBBAC8C 2A9ABBE7 53F52259 A7B7D70E
xR31 =
0019E75F 0F03312D 22CBBF15 3747525D 89E5155B ABB8BF0C 130CB567 CA532F69 AAF57EA7 682B9957 021D9041 4433ABBE EDC233E9 08218578 1C16724C 8C356777

3.2.2 SIAKE751

p =
00006FE5 D541F71C 0E12909F 97BADC66 8562B504 5CB25748 084E9867 D6EBE876
DA959B1A 13F7CC76 E3EC9685 49F878A8 EEAFFFFF FFFFFFFF FFFFFFFF FFFFFFF-
F FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFF-
F FFFFFFFF
e2 =
00000174
e3 =
000000EF
xQ20 =
00003E82 027A38E9 429C8D36 FF46BCC9 3FA23F89 F6BE06D2 B1317AD9 04386217
83FDB7A4 AD3E83E8 6CAE096D 5DB822C9 8E561E00 8FA0E3F3 B9AC2F40 C56D6FA4
A58A2044 9AF1F133
5661D14A B7347693 63264608 6CE3ACD5 4B0346F5 CCE233E9
xQ21 =
00000000
yQ20 =
00003BBF 8DCD4E7E B6236F5F 598D56EB 5E15915A 755883B7 C331B043 DA010E6A
163A7421 DFA8378D 1E911F50 BF3F721A 8ED5950D 80325A8D 0F147EF3 BD0CFEC5
236C7FAC 9E69F7FD 5A99EBEC 3B5B8B00 0F8EEA73 70893430 12E0D620 BF-
B341D
yQ21 =
00000000
xP20 =
00005492 1C31F0DC 9531CB89 0FC5EC66 DF2E7F0D 55761363 C6E375DA 69B0682C
ABE5C0FF FCBE6E1A D46563F0 42FA06B9 F207FCF3 CDD26736 52828FF5 0C3F7B75
5C0BE072 950D16CA
747C1467 75C0267A 401FFC73 8B03A49E 9A36B395 72AFB363
xP21 =
00002884 9BC0D81E 01993137 A5B63D6E 633C4E97 AB4FF118 CCF63DFE 623092AC
86B6D4A9 B751797C BA1A1775 00E9EB5A F7852B7D F02C3348 44D652EF C4729178
A1DBAD8C A47BB7E7 57C6D43B 799811A6 3BEBE649 C18101F0 3AD752CD CD73BF66
yP20 =
00001961 19D87272 DC3AA722 3476C8C3 269D48CA EFAE692F 68DCF2D6 E1BEB5B9

7525D502 6C157C7C 740B41AD E80A8CF2 E1E0B37E 5F5FD4ED 88235BF7 404BE391
89C137E2 1C035EF6 339D7FAC BA38E72D 69043710 E76266A5 FC14EFB9 5E5FBC7C
yP21 =
0000D3AC 09A67D59 CC8D78B0 FA6681AE 78BDF0C8 F558E386 6005E435 5B0B1993
18D9CDD6 7C0A7DB2 34F9EA1E C4C5F1E5 9168B7DB D14281F0 9E8DF904 A3D574CA
D526DC5A 3667490A DE1A4C13 B09F7B11 5C4E488F D4DD5F76 70B58973 22AD41D
xR20 =
000022A0 B5A35A2B 0C56135A 7CEC5CFB 97964A7C 6226FE90 9F374362 A8ECA3AB
14A1B7B0 C87AC875 DCE5888D 83B623BF 0011A4AC 138F62EF 6B2D2D84 F636548A
9F920F23 8336E5A3 6E45E405 5940E3C9 4385B8FC 53743964 32EEF2AE 178CEFD-
D
xR21 =
00000F9C 4AFCDA80 9C3358B0 96B250C6 9B20310F DF2EF631 711AA4EF EC49A4E7
6483F320 B793F2EB C63365EE D14AA3F6 EA33FEB5 6796F011 BA6C6DFB 4D0A00AA
C4D27866 46D914AD 026CBB4A 592EC74B 5485372E 51382D44 528DD491 B83D9547
xQ30 =
00002F1D 80EF06EF 960A01AB 8FF409A2 F8D5BCE8 59ED725D E145FE2D 525160E0
A3AD8E17 B9F9238C D5E69CF2 6DF23742 9BD37786 59023B9E CB610E30 288A7770
D3785AAA A4D646C5 76AECB94 B919AEED D9E1DF56 6C1D26D3 76ED2325 DC-
C93103
xQ31 =
00000000
yQ30 =
00000127 A46D082A 1ACAF351 F09AB55A 15445287 ED1CC55D C3589212 3951D4B6
E302C512 9C049EEB 399A6EDB 2EEB2F9B 0A94F06C DFB3EADE 76EBA0C8 419745E9
7D12754F 00E898A3 15B52912 2CFE3CA6 BBC6BAF5 F6BA40BB 91479226 A0687894
yQ31 =
00000000
xP30 =
000005FD 1A3C4DD0 F6309741 96FED351 9152BC70 98B9E2B1 21ECA46B D10A5CC9
F4BCC6C6 89B8E4C0 63B37980 75FCEE6E DAA9EB10 8B3CD004 95CF04DD 8CE4A08F
BE685A12 7D40E45F 4CF45098 A578DEB4 43686993 94C43BFC 9BC5E000 52F78E8D
xP31 =
00002B88 A03360B3 38954773 2C9140C0 5DEA6516 881FE108 211BE887 CC43FCB8
0C06A1D8 6FF5457D 3BB7DB93 6394EC33 821AA393 33A60AF8 4B537974 CFA0BA82

87D699D2 BF79BA55 9026C64A 6ED61050 1D2357C1 0B9A6C8F 83742492 2275ACBF
yP30 =
000053B5 5053E3F0 4FC315EF B1B7B2C4 AFCB4FEF 12CE744A F3B243C6 E6B1417E
94A78D49 80DDE181 89646492 3E01AACC 3DA040A0 747CA675 54A35268 4DA207C4
9022D930 732DF6BD 0BF37E1F 5C169176 69A70F88 059C1C73 9A79D7CF A0C529D9
yP31 =
0000044E 44196909 252ECD7B 91643238 15294F02 AED22C4E 4EB43D2C E2BC5F29
EB575D45 CA8B6B4C 4242E369 AE3A1EFC 844E9D1C 57B0AE33 74BC2CED AD16B0C6
99158332 E2D9AB3F 0025C034 8C5F70FD C4DD7C48 65E64B8B 843F03D8 07447D5E
xR30 =
0000077B 3BB69009 428A327D 43CA6016 9715F547 454F88CD 017B32DF 58A7252C
2B3C3D00 D52CCD31 33D54041 D8BCAEA2 91F20572 02328712 CD395575 CD7CCD3C
E70C0A1E BF633BA9 46559458 878F41F9 FDD1727E 2C31125B 2FE5B713 06704829
xR31 =
00006D91 393A57DB F47FD6DC F841F17E CD719CAE 1D33C683 2A75B0F1 68855BC-
C 38D2A479 2DFF9BC8 6DEACA10 B1AA808D 539B167D 73BBA321 68687FA3 F85AE93A
1ADDE5BD 1FD5B681 DCC6C344 54D44969 76C22D80 C95E42B1 2576FC0F B4074B9F

3.2.3 SIAKE964

p =
00000008 6B5BFF76 43C64F7A 10028248 AD4FC4B1 50CBAA75 A2A1FA44 CBAB2451
35469BAB 093F2B8D AD5281E7 E56EF6AA 57A94749 ABB38EAB 467ACDE5 451CD4BF
FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFF-
F FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFF-
F FFFFFFFF
e2 =
000001E6
e3 =
0000012D
xQ20 =
00000001 CD5AEB4E 02DBE2CE B712A45E ED7720D3 EA94116F 1E45C834 FDF-
F3A86 7BBB267B F8F5F9B1 9C369F7A FE141B85 D591243E 7310B6D0 2E78DB88
8615254D F178C1F7 5F2BDAF7 03E83BB9 7DBCDC3D FDB60BA3 85EC8F42 D4AD1505
21ECA6EC 4D3086A7 783698A7 1544E10A 45EA605E 1B86A894 7F14FA2E 03845DAE
xQ21 =
00000000
yQ20 =
00000002 2E751F1F 60841CF4 E8D4D3BD 8D400F58 9761CA1F 71A9C1F3 83C0FE55
3E6492DB E1F78D5F 9A768920 B682786E 8125398F 765A481B 32913561 FD16B270
19D9C10C 4F9062AC 1513FEB2 FE942DD2 2AC53F6E C319C4D1 8A53A481 430F3DFA
22E57EDA 0D067C37 F91EA8F1 3E4B1C65 4E974856 781F8E0A 397ED362
yQ21 =
00000000
xP20 =
00000006 ED767E28 04975D81 80368FB9 A72CE64E 838A5497 4865BFF1 A86AEF07
D6171A8A 4DF351F1 D4C94AAF 82BD6EBD 396F3342 48282F50 73178AB5 7B906BEF
89A2A152 A10D04A5 B20A0FFF 96B0B48F 0599FC9B D2AD52E0 81BB7FEA 7B5E8BF4
C3B0AB13 0731F4C5 A974CFA5 AD678121 7A20F9EC D30691D9 D1941D03
xP21 =
00000003 FE63FBDC A589518A 3DA694EC C8B65934 6693C45B D8AC86B6 F0C778CF
290C9F42 9163FEF6 4AFAD182 ADE1B0C4 DFC8CF29 C35455C7 BA69C225 59F2E0D4
20AE05BB 0AE3ADC0 9A4A0AF2 8CE1A1C5 93171033 7AA68884 EFCCD60A C76FD3F1
7ED50205 305509E0 5955F60D 5008D788 B83F5FA4 57FD79EC DC7179D1

yP20 =
00000000 4E0A8662 85403BC0 408F9BCA 912025E3 17111896 1C865461 2AE20CEE
AF91A98A 4F278EAE BB704602 8AD90CD9 5B99BF6B 34233CD4 B084B2CA 4598D-
F3A 6D4839CA 6EA493ED 420CF4C1 3A1F37F1 EC59620F 08693649 C72380A8 479E3753
93D3F4A7 59DF65F1 F74B4C65 6B79DF2A 5DA2959E FB006BDA D015D252
yP21 =
00000007 38846048 2319281B 78C6AC1F E1A91DB7 2A2C9AA3 4BEE1EBE A33EF043
AA1BFA0C 45894142 95E94C91 1E19E808 246B2A0A 98593958 E1F70888 E00332DE
AE7D7FDB CA53398E 59530E5D 2A292463 7533F46C 4684373D DDB8D09B 2A75C307
3EA3C19E CF946FCB 2B428B6E 9CF93F22 33DD257C 5CAD4041 3F78CD1D
xR20 =
00000008 44F024B8 D993B660 48C9DA7F 1724AE2E 4C6162F8 4804FE3F E290FBEB
5ABF7DF2 5C395121 77C8E4A4 7A35F8EC D037B699 E34F58EF 675AE188 A1537838
A4DDBA69 FC7BC3FA CB7E3815 F3031244 AF1BCCE1 95AF45B4 2A587EE8 7A00BF2D
A1E972D8 D662F4DC 5EC1CDB1 03D9EED2 215D4DD7 48004985 8925A63A
xR21 =
00000007 35F06B97 66B69FF9 17835EAD C539A00F FC186ED2 5947F701 FDA7EDEA
F517039F 9B0DA172 1FDAE978 4838C75B 46A452DC 902EC8DD 1F462564 3B42C596
C2CF0404 5A7AA804 3F07C9A9 F82611D2 02F06834 512A3803 EF64650E 5309163F
25CCC336 CC852764 9E340F59 CDDFB51B 24D1C02E 8CB2653E 7A05B709
xQ30 =
00000003 81DBCAB1 EE7A4CA3 192CDA85 3F4E0F42 6522EB9D 3277421C 29D73CC4
F70BEFE7 009767C4 AE451600 3B237223 422C0E75 2ACD9D8F CE07263D 2C1D1013
08C0B97E DB8D4A8C 53C2064B 05DF9A61 E8216CA1 FFAC55F4 CA043972 52704945
C27136A0 56F6B5CF 8838B7F6 52BC16C1 392B5597 36CAF63B F0058A53
xQ31 =
00000000
yQ30 =
00000001 AFBFA81B 55C7D789 B6E89BC7 A311F3CE E4B733B0 FA5B7D56 D29A644B
596B7729 778E0773 F908D76E 0377B3CA 41C03D79 7DE4F0B7 985EB512 7D2151EC
4B6C1136 1AEB4CEA A3F776E0 8E4AD01B 7BB46074 D425C8A2 61E88B14 5C4153BF
67732E82 9986EB9D 29C88385 1EEEB87C C4FD96A0 84332542 6C108687
yQ31 =
00000000
xP30 =

00000004 FE46F0B0 09171C87 0FE840B7 FD0C3F14 93813CD1 25C2191C 9FA4BDE4
0941A603 124F1B81 BBFBEBFD AC06F808 07562639 FC61A579 62AE6E6B 7EE793CF
7B359746 FEB0DA11 0D704681 F83EF6B4 40DC5DED D2A42471 49D0A44C 452ED374
A394319A 8888A2A0 9CC4A0F3 5A07AA3D 248CF780 3E77EAD2 A4BEA308
xP31 =
00000006 DD4FC176 8E1ADEE5 4DC4E41C FAA7B810 4644DD69 0616D374 A9139013
FD847C2C D11BA6CA 4C4FC26A 63FE198B 666B7912 FBF889E9 91CF4B90 3651F441
4CD4AA50 BC02CE2E C986A7BF C1A8D364 F93410AD E3B959FF 1F036F36 E-
F3AFF88 D28DB500 8276C340 2158ADB4 A44BAECE D2AF6503 093FD8B6 A58EC136
yP30 =
00000003 EA8CDCF4 BC1C1B9D 2D449022 387D4DDE F05CE98C B63E722B 0EA14717
C5FFA82E E107832E 5FE58C28 90185D2C 90D1BD94 AC13C69F D483AC80 66B1F1A4
844F7655 884B2379 0088A6DA 915FD709 EFE79A88 028108F4 D4DFBFAD BA65EFBB
C5D621BA 31F12BE6 FB717D3B 1D8CD78D CD05B0D2 B7E87C10 3D1897C0
yP31 =
00000007 77607A21 85C04FBF CFA5EAF7 7F38F40F 42746739 748CA176 BBE31739
4BDA28F3 D971DFB9 CCB67207 E201FFB0 0A9A3E6B 9B7E804F BB6EF61E CEB-
DB8AC 68831E10 E8A72613 A47F132B D9A2309E 404FCFFF 7EA7BC87 AD448B8B
8798AB61 CA6F97DB 3B240887 9DEB8A9F 930C4EE4 69486FA1 129E89B6 7C084CD9
xR30 =
00000007 DE290085 EBBDC801 A1D6292D 1F2E89FF 463669ED 2F5F6C02 B8010A75
245C4D39 84002821 B8A243C7 56512A5F C1FC0867 A84583D7 6B0404E7 E73CEB70
71E2AE3B F43BFB77 A87BC98F DF888E28 5CD4A3C9 4E4D1795 009E41ED B8A3AD8F
81321138 E4A87B69 416AEFF0 94E541F4 8681863B AD30FB2F 32EA019A
xR31 =
00000007 A2A2DFA4 FB567336 60ACCB80 308A4482 B1A46D3B 9BB20313 F164CC80
9A3A6B4D 2FBC4357 4994354D C06D409F 9F647E82 F4F05D6C 5A70E340 DF4B9555
9787F82E AD7F7295 590FDCD9 D54B8001 094DC809 29EF4C5A BB8E388A 53AA0BD3
88D890B5 980F1FD1 9404025B 582C640D DFDA1BDF 46D37046 4A812732

4 安全性分析

4.1 经典和量子模型下抵抗攻击类型

SIAKE 算法可以抵抗认证密钥交换协议中常见的攻击手段，包括CK$^+$安全模型中的所有攻击情形.并且我们分别证明了方案在经典随机预言和量子随机预言模型下的安全性.

4.1.1 经典随机预言模型

我们在经典随机预言模型下将攻击 SIAKE 的困难性转化为解决同源上的判定 SIDH 问题的困难性. 具体如下定理：

定理4.1 (见 [XLL+18] 定理1, [XXW+19] 定理2). *在判定 SIDH 问题困难的假设下, SIAKE 在经典随机预言模型下是 CK^+ 安全的(见定义 1.3)，并且支持用户任意注册公钥. 严格来说，如果协议中有 N 个用户，而且两个用户间最多有 l 次会话, 对于任意多项式时间 CK^+ 敌手 $\mathcal{A}$, 存在敌手 $\mathcal{B}$，其优势最多为*

$$\mathbf{Adv}^{\mathrm{CK}^+}_{\mathsf{SIAKE}}(\mathcal{A}) \leq N^2 l \left(4Adv^{siddh}_{\mathcal{B}} + \frac{q}{\max\{l_1^{e_1}, l_2^{e_2}\}} \right),$$

其中 q 为进行 CK^+ 访问的次数.

该定理和我们的工作 [XXW+19] 中的定理 2 相同，同时也是我们的工作[XLL+18]中的定理 1 与定理 7 的组合，严格的证明可参见两个附件，此处简单解释：根据 [XXW+19] 中 Lemma 2 的结论和证明本文中 2.1 节中的 2-Key PKE 为 [OW-CPA, OW-CPA] 安全的. 而 [XLL+18] 中的定理 1 与定理 7 保证如果 2-Key PKE 为 [OW-CPA, OW-CPA] 安全的则 SIAKE 在经典随机预言模型下CK+安全. 结合引理 2.1, 可得该定理.

4.1.2 量子随机预言模型

另外，我们在 [XAY+19] 的工作中证明了 SIAKE 可以在量子随机预言模型下将方案的 CK$^+$安全性归约到判定 SIDH 问题困难假设上. 具体需要底层 2-key PKE 的分布可计算的属性.

定理4.2 (见 [XAY+19] 定理 1 以及 [XAY+19] 5.1节的分析). *在判定 SIDH假设下，以及 $2PKE_{sidh}$ 可分步计算的属性下,AKE_{QRO} 为量子随机预言模型下是 CK^+安全的. 具体来讲，假设 AKE_{QRO}有 N个用户，并且每两个用户之间最多有 l 次会话. 对于任意多项式时间的的量子敌手$\mathcal{A}$，其最多访问q 次CK^+ 询问, q_h (resp. q_G, q_H, q_f, $q_{h'}$) 次对h*

(resp. G, H, f, h') 的量子询问, 则存在判定 SIDH的解决算法$\mathcal{C}$ and $\mathcal{D}$, s.t.,

$$\begin{aligned} Adv_{\mathcal{A}}^{ck+}(n) \leq \quad & 2N^3 l^{3/2}(q_G + q_h)\sqrt{2Adv_{\mathcal{C}}^{siddh} + 2^{-n} + 2^{-(k-n)/2}} \\ & + N^2 l \cdot (q + q_H)2^{\frac{-n+1}{2}} + N \cdot (2q + q_{h'} + q_f)2^{\frac{-n+1}{2}}, \\ Adv_{\mathcal{A}}^{ck+}(n) \leq \quad & 2N^3 l^{3/2} q_G \sqrt{2Adv_{\mathcal{D}}^{siddh} + 2^{-n} + 2^{-(k-n)/2}} + N^2 l \cdot (q + q_H)2^{\frac{-n+1}{2}} \\ & + N \cdot (2q + q_{h'} + q_f)2^{\frac{-n+1}{2}}. \end{aligned}$$

该定理严格证明同我们的工作 [XAY+19] 中的定理 1 以及 [XAY+19]中第 5.1 节的分析，此处略过.

4.2 抵抗攻击类型

具体来讲，以上两个定理说明，SIAKE不仅考虑了经典的CK敌手攻击能力还考虑如下攻击类型：

- **支持任意注册**：敌手可以任意注册公钥，甚至复制其他人的公钥，而不需证明其拥有合法的相应私钥. 可信的CA也不用验证注册用户公钥的合法合理性.
- **弱前向安全性**: 支持弱的前向安全性，也就是如果之前的目标会话是敌手被动监听和记录的，即使用户 A和 B的长期私钥都泄漏给敌手，会话密钥仍然安全.
- **KCI安全性**：抵抗密钥泄漏伪装攻击.
- **MEX安全性**：最大泄漏安全性指敌手获取用户 A和用户 B的所有随机数，其之间的会话密钥仍然是安全的.
- **侧信道攻击**：由于在 CK$^+$安全模型中，考虑敌手通过侧信道攻击获得用户的内部状态，随机数或者长期私钥，SIAKE抵抗敌手侧信道攻击获取部分信息.

4.3 实际攻击算法的复杂度

由定理 4.1和 4.2,我们将对于 SIAKE 的各种攻击优势转化为攻击底层判定 SIDH 问题的优势. 因此在分析攻击 SIAKE 困难性上我们主要分析解决判定 SIDH问题的复杂度.

在文献[UJ18](定理4.8) 中指出判定 SIDH问题和计算的 SIDH问题是等价困难的. 在同源类计算中有一个同源的随机游走问题(supersingular isogeny walk) [JAC17] 是指给定两个同源类的曲线E 和E' 找到一个E 到E' 的一个同源路径问题. 同源游走问题一定比判定SIDH问题困难. 针对同源随机游走问题目前最好的解决算法仍然是 Galbraith [DG16, Gal99] 的中间相遇解决办法.

针对计算 SIDH问题的复杂性分析采用 [JAC17] 文档中的说明，如下. Galbraith [DG16, Gal99]方法可以扩展到攻击计算 SIDH问题上. 对于 $\mathbb{F}_{p^2}$ 上的超奇异同源类的大小大约为 $p/12$,采用中间相遇解决办法，在 $\mathcal{O}(\sqrt[4]{p})$复杂度下可以计算 E_0 到 E_B 的之间的阶为 2^{e_2} 同源映射. 然而如果使用量子算法，效果更好. 使用 Tani [Tan09] 的 claw-finding 算法, 只需要在 $\mathcal{O}(\sqrt[3]{2^{e_2}}) \approx \mathcal{O}(\sqrt[6]{p})$复杂度下可以计算 E_0 到 E_B 的之间的阶为 2^{e_2} 同源映射.

以上两个算法是目前效率最好的经典算法和量子算法，解决困难问题的复杂度分别为 $\sqrt[4]{p}$和 $\sqrt[6]{p}$. 但是由定理 4.2, 在量子随机预言模型下， SIAKE 的安全性和底层困难问题的安全性有一个平方根的差距，因此在量子随机预言模型下，其量子复杂性的安全级别降低一半. 因此，SIAKEp503, SIAKEp751 和SIAKE964 三组参数下的具体安全性如下表所示，

参数			经典RO下	量子RO下
		经典复杂度	量子计算复杂度	量子计算复杂度
SIAKEp503	128	2^{125}	2^{83}	2^{41}
SIAKEp751	192	2^{186}	2^{125}	2^{62}
SIAKEp964	256	2^{238}	2^{160}	2^{79}

表 2: 三种参数下的安全级别与复杂度. 其中RO为随机预言的简写.

5 性能分析

本节我们给出在三种安全参数下，长期公私钥的尺寸、通信量大小以及计算复杂度.

5.1 长期公私钥尺寸

参数	Public A	Secret A	Public B	Secret B
SIAKEp503	378	32	378	32
SIAKEp751	564	47	564	48
SIAKEp964	726	61	726	61

表 3: 三种参数下的长期公钥私钥尺寸. 单位为Bytes.

5.2 通信量

参数	$A \to B$	$B \to A$	总通信
SIAKEp503	788	442	1214
SIAKEp751	1160	628	1788
SIAKEp964	1484	790	2274

表 4: 三种参数下通信量大小. 单位为 Bytes

5.3 计算复杂度

我们在 Intel 酷睿 i7-6500U 2.50GHz 处理器，8GB 内存，Windows 64 位操作系统下，使用 Microsoft Visual Studio 2017 对 SIAKEp503和SIAKEp751 两种参数下的算法进行测算.

其中参考实现的复杂度如表 5 所示. 参考实现的具体运行时间如表 6 所示.

优化的 x64 实现的复杂度如表 7 所示. 优化的 x64 实现具体运行时间如表 10 所示.

我们在 Intel 酷睿 i7-6500U 2.50GHz 处理器，8GB 内存，Windows 64 位操作系统下，使用 Microsoft Visual Studio 2017 对 SIAKEp964 参数测试. 优化的 x64 实现的复杂度如表 9 所示. 具体运行时间如表 10 所示.

参数	SIAKE.Reg A	SIAKE.Reg B	SIAKE.A.int	SIAKE.B.Shared	SIAKE.A.Shared
SIAKEp503	1001456	1105634	2863245	5622456	28145264
SIAKEp751	32879127	3705468	9619023	16997982	9222421

表 5: 参考实现(release+win32)计算复杂度. 单位10^3 Cycles. 该值为循环10000次后计算每次的平均值.

参数	SIAKE.Reg A	SIAKE.Reg B	SIAKE.A.int	SIAKE.B.Shared	SIAKE.A.Shared
SIAKEp503	187	214	527	860	438
SIAKEp751	497	564	1535	2641	1702

表 6: 参考实现(release+win32)计算时间. 单位ms. 该值为循环10000次后计算每次的平均值.

参数	SIAKE.Reg A	SIAKE.Reg B	SIAKE.A.int	SIAKE.B.Shared	SIAKE.A.Shared
SIAKEp503	16965	19165	47423	84654	45867
SIAKEp751	52378	62445	151387	272965	147125

表 7: 优化的x64(x64+release)实现计算复杂度. 单位10^3 Cycles. 该值为循环10000次后计算每次的平均值.

参数	SIAKE.Reg A	SIAKE.Reg B	SIAKE.A.int	SIAKE.B.Shared	SIAKE.A.Shared
SIAKEp503	7	9	23	43	24
SIAKEp751	21	23	60	108	58

表 8: 优化的x64(x64+release)实现计算时间. 单位ms. 该值为循环10000次后计算每次的平均值.

参数	SIAKE.Reg A	SIAKE.Reg B	SIAKE.A.int	SIAKE.B.Shared	SIAKE.A.Shared
SIAKEp964	2566474	2948823	7754798	13261345	7456358

表 9: 优化的x64+release实现计算复杂度. 单位 10^3 Cycles. 该值为循环10000次后计算每次的平均值.

参数	SIAKE.Reg A	SIAKE.Reg B	SIAKE.A.int	SIAKE.B.Shared	SIAKE.A.Shared
SIAKEp964	1029	1096	3032	5024	2813

表 10: 优化的x64+release实现计算复杂度. 单位 ms. 该值为循环10000次后计算每次的平均值.

6 优缺点声明

SIAKE 的最大优点就是考虑了认证密钥交换协议中几乎是最强大的敌手攻击能力，提供了弱前向安全性、任意注册、KCI安全性和MEX安全性; 可以提供量子随机预言模型下的安全性；具有极低的带宽通信量；任意底层同源计算和曲线参数的改进和优化都可以应用到该算法中. 同时该算法最明显的缺点就是计算量大，相对于同等级别的格算法来说计算复杂度相当于百倍左右. 在如下章节详述.

6.1 优点

- 通信带宽极低. 在 128 安全级别的参数下通信量仅为 800 bytes以下，非常适配现有的通信协议；192 安全级别下通信量也控制在 1.1K以下;
- 无解密错误；
- 考虑了认证密钥交换协议中基本上所有的可行攻击下的安全性，包括弱前向安全性、任意注册、KCI 安全性和 MEX安全性;
- 同时提供了经典随机预言和量子随机预言的安全性;
- 算法底层安全参数和曲线都极容易替换，任意在同源计算上的改进都可以应用到此,例如压缩技术 [CJL17]和快速计算的技术 [FLO17];
- 在椭圆曲线基础上改变的,容易经现有基于椭圆曲线的算法进行迁移.

6.2 缺点

- 计算效率低;

参考文献

[BDK+11] Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, Franois-Xavier Standaert, Yu Yu, Leftover Hash Lemma, Revisited, In CRYPTO 2011, pp. 1-20

[BR93] Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS 773, pp. 232-249. Springer, Heidelberg (1994)

[CJL17] Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient compression of SIDH public keys. In EUROCRYPT 2017, pp. 679-706.

[CLN16] Craig Costello, Patrick Longa, Michael Naehrig: Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. CRYPTO (1) 2016: 572-601

[CK01] Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453-474. Springer, Heidelberg (2001)

[DG16] Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over Fp. Designs, Codes and Cryptography, 78(2):425 – 440, Feb 2016.

[FLO17] Faz-Hernádnez, A., López, J., Ochoa-Jimenez, E., Rodriguez-Henriquez, F.: A faster software implementation of the supersingular isogeny diffie-hellman key exchange protocol. IEEE Transactions on Computers (2017).

[FO99] Fujisaki, E., and Okamoto, T.: Secure Integration of Asymmetric and Symmetric Encryption Schemes. In: Wiener M. (eds) CRYPTO 1999, LNCS 1666, pp. 537-554. Springer, Heidelberg (1999)

[FSXY12] Fujioka A., Suzuki K., Xagawa K., Yoneyama K.: Strongly Secure Authenticated Key Exchange from Factoring Codes and Lattices. In: Fischlin M., Buchmann J., Manulis M. (eds) PKC 2012, pp. 467-484. Springer, Heidelberg (2012)

[FSXY13] Fujioka A., Suzuki K., Xagawa K., Yoneyama K.: Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In AsiaCCS 2013, pp. 83-94.

[FTTY18] Fujioka, A., Takashima, K., Terada, S., Yoneyama, K.: Supersingular Isogeny Diffie-Hellman Authenticated Key Exchange. IACR Cryptology ePrint Archive 2018/730.

[Gal99] Steven D. Galbraith. Constructing isogenies between elliptic curves over finite fields. LMS Journal of Computation and Mathematics, 2:118 – 138, 1999.

[Gal18] Galbraith, S. D.: Authenticated key exchange for SIDH. IACR Cryptology ePrint Archive 2018/266.

[GPST16] Galbraith, S. D., Petit, C., Shani, B., Ti, Y. B.: On the security of supersingular isogeny cryptosystems. In ASIACRYPT 2016, pp. 63-91.

[HHK17] Hofheinz, D., Hövelmanns, K., and Kiltz, E.: A Modular Analysis of the Fujisaki-Okamoto Transformation. In Y. Kalai and L. Reyzin (eds) TCC 2017, LNCS 10677, pp 341-371. Springer, Heidelberg (2017)

[HKS+18] Hofheinz, D., Kiltz, E., Schäge, S. and Unruh, D.: Generic Authenticated Key Exchange in the Quantum Random Oracle Model. eprint archive: report 2018/928.

[JD14] De Feo, L., Jao, D., Plut, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. Journal of Mathematical Cryptology, 8(3), 209-247 (2014).

[JZC+18] Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, Zhi Ma: IND-CCA-Secure Key Encapsulation Mechanism in the Quantum Random Oracle Model, Revisited. CRYPTO (3) 2018: 96-125

[JAC17] Jao, D., Azarderakhsh, R., Campagna, M., et al: Supersingular Isogeny Key Encapsulation. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions.

[Kra01] Krawczyk, H.: The order of encryption and authentication for protecting communications (or: How secure is SSL?). In: Kilian J. (eds) CRYPTO 2001, LCNS 2139, pp. 310-331. Springer, Heidelberg (2001)

[Kra05] Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (eds) CRYPTO 2005. LNCS, vol. 3621, pp. 546-566. Springer, Heidelberg (2005)

[LLM07] LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1-16. Springer, Heidelberg (2007)

[Lon18] Longa, P.: A Note on Post-Quantum Authenticated Key Exchange from Supersingular Isogenies. IACR Cryptology ePrint Archive 2018/267.

[ML81] Mark N. Wegman and Larry Carter. New hash functions and their use in authentication and set equality. J. Comput. Syst. Sci., 22(3):265 – 279, 1981.

[Tan09] Tani, S.: Claw finding algorithms using quantum walk. Theoretical Computer Science, 410(50), 5285-5297 (2009).

[UJ18] Urbanik, D., Jao, D.: SoK: The problem landscape of SIDH. IACR Cryptology ePrint Archive 2018/336.

[Vel71] Jacques Vélu. Isogénies entre courbes elliptiques. CR Acad. Sci. Paris Sér. AB, 273:A238 – A241, 1971.

[XLL+18] Haiyang Xue, Bao Li, Xianhui Lu, Bei Liang, Jingnan He: Understanding and Constructing AKE via Double-Key Key Encapsulation Mechanism. In: Peyrin, T., Galbraith, S. (eds.): ASIACRYPT 2018, LNCS 11273, pp. 158 – 189, 2018. Springer, Heidelberg (2018) 见附件1

[XXW+19] Xiu Xu, Haiyang Xue, Kunpeng Wang, Man Ho Au, Song Tian: Strongly Secure Authenticated Key Exchange from Supersingular Isogenies. In AISACRYPT 2019. 见附件2

[XAY+19] Haiyang Xue, Man Ho Au, Rupeng Yang, Haodong Jiang: Compact AKE in the Quantum Random Oracle Model, 未发表. 见附件3