SCloud
基于 LWE 的加密与密钥封装算法

郑中翔*, 王安宇†, 樊海宁*, 赵春欢* , 刘超‡, 张雪*

*清华大学

† 中国科学院信息工程研究所

‡ 山东大学

October 20, 2019

目录

1 介绍以及设计思路

我们的提交方案包括公钥加密方案与密钥封装方案，加密方案包括 SCloud-CPAPKE与 SCloudCCAPKE，其中 SCloudCPAPKE 可实现 IND-CPA 安全，密钥封装方案 SCloudKEM 将 SCloudCPAPKE 与哈希函数作为内部构件通过FO转换实现 IND-CCA 安全，进一步地， SCloudCCAPKE 为使用 SCloud-KEM 构造 IND-CCA 的加密方案。我们使用当下被广泛研究的 LWE 困难问题来设计方案。因为 LWE 困难性基于无代数结构格上的困难问题，所以安全强度较有代数结构的困难问题（RLWE,MLWE）要高，近几年被广泛用于方案设计。

1.1 设计思路

在抗量子方案设计方面，基于格的密码体制是较为有优势的一类。格是离散的加法群，格上的核心计算困难问题是最短向量问题（SVP）和最近向量问题（CVP）。格上各种操作的运算速度是相对较快的，且至今的研究表明没有量子算法能够有效求解格上困难问题。目前，NTRU 和基于 LWE 及其变体的格密码体制是最成熟、应用最广、效率较高的两类格密码体制。 LWE 问题的困难度是基于一般格上最坏情况的困难问题，我们选择基于 LWE 问题设计体制。

无结构格 vs 带结构格 LWE 问题[31] 是寻找含扰动线性系统解的问题，也可以被转化为求解一个无代数结构随机格上的 SVP 问题。RLWE[27, 30], MLWE[11, 24] 以及一些特定 NTRU[33] 问题基于的是有代数结构格的困难问题，他们的量子复杂度分别被证明不低于其对应的理想格或模格上最坏情况困难问题[27, 24, 30]。相较于基于 LWE 问题的方案，基于带结构格困难问题的方案有显著的性能优势，所以 RLWE, MLWE, NTRU 等 LWE 变体被广泛应用于快速方案的设计。

从安全性角度来看，基于带代数结构 LWE 变体的方案在推荐参数下能抵抗已知攻击，并没有明显漏洞。但是目前的一些研究表明带结构格可能存在一些隐患，例如对 NTRU 方案基于的带结构格存在特殊攻击[22, 23]，在该攻击下其渐进复杂度显著低于相同参数下一般格的复杂度。最近的工作指出，在非常广

泛的一类环对应的理想格上，近似参数为亚指数$2^{\tilde{O}(\sqrt{n})}$ 的最坏情况 SVP 问题存在量子多项式时间算法[14, 15]，而对于一般格来说，仅有亚指数$2^{O(n \log\log n/\log n)}$时间算法[32]。若要攻破基于理想格困难问题的方案，需要近似参数为多项式的算法，且 RLWE, MLWE 等问题本身的困难度为不低于理想格上的困难问题，所以现有分析不影响当下众多（安全度基于带结构格的）方案的安全度。

当我们设计后量子密码体制时，首先需要长达数年的时间来设计与研究一个成熟的密码体制，并且我们期望其在未来足够长的一段时间内是安全的。将未来量子计算机与量子算法的发展考虑进来，我们的思路是尽可能保守地设计后量子密码体制，以保证方案在未来几十年内安全，而不是过早地使用当下性能最优的设计思路。为此，我们的方案基于无代数结构格的 LWE 问题，而不是有代数结构的困难问题（RLWE, MLWE等）。我们的方案虽然在公私钥规模与运行效率上表现比RLWE,MLWE等方案稍差，但本方案达到的规模及效率在目前的应用环境下仍然是高效快速的。

创新点 方案主要有两个创新点：采样与纠错码。方案引进了一种新型的采样方法，该方法可以看作二项分布与有界均匀分布的卷积。通过选取合适的参数，该采样方法对比已知存在的采样方法可实现更高的效率和安全强度。同时，我们设计了高效的纠错码方案。与之前常用的简单的纠错码方案相比，该纠错方案充分利用了加密空间，其不仅可以减小参数规模，改进加解密效率，而且使参数设置更加灵活。基于上述两种创新，我们的方案参数规模与效率对比Frodo更优。

文档安排 我们在 Sect.2 给出文档中需要用的基础知识，在 Sect.3 中给出我们的算法描述，Sect.4 给出我们的方案参数，安全性分析放到 Sect.5，在 Sect.6 给出方案的性能分析，在 Sect.7 给出适配性说明，最后在 Sect.8 给出方案优缺点声明。

2 准备工作

2.1 符号表示

首先，我们给出本文中用到的基本符号表示：

- 用粗体的小写字母表示向量（如：$\mathbf{a}, \mathbf{b}, \mathbf{v}$），用粗体的大写字母表示矩阵（如：$\mathbf{A}, \mathbf{B}$）。对于给定的集合 D，元素属于集合 D 的 m 维向量的全体记为 D^m，元素属于集合 D 的 $m \times n$ 阶矩阵的全体记为 $D^{m \times n}$。
- 对于 n 维的向量 $\mathbf{v}$，其第 i 项元素表示为 $\mathbf{v}_i$ $(0 \leq i < n)$。
- 对于 $m \times n$ 阶矩阵 $\mathbf{A}$，其 (i, j) 项（即第 i 行中第 j 列的元素）表示为 $\mathbf{A}_{i,j}$（$0 \leq i < m$ 以及 $0 \leq j < n$），矩阵的第 i 行表示为 $\mathbf{A}_i = (\mathbf{A}_{i,0}, \mathbf{A}_{i,1}, \cdots, \mathbf{A}_{i,n-1})$。
- 对于 m 比特长的字符串 $k \in \{0,1\}^m$ 可以看作是集合 $\{0,1\}$ 上的向量，其第 i 比特（$0 \leq i < m$）元素表示为 k_i。
- 整数环用 $\mathbb{Z}$ 表示；对于正整数 q，整数模 q 的商环用 $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ 表示。
- 对于概率分布 χ，符号 $e \leftarrow \chi$ 表示根据分布 χ 采样获得 e。χ^n 表示 n 个相互独立的 χ 概率分布。
- 对于有限集 S， S上的均匀分布用 $U(S)$ 表示。
- 小于或等于 a 的最大整数用 $\lfloor a \rfloor$ 表示；距离 a 最近的整数用 $\lfloor a \rceil = \lfloor a+1/2 \rfloor$ 表示。
- 对于实向量 $\mathbf{v} \in \mathbb{R}^n$，其欧几里德范数（即：2 范数）用 $\|\mathbf{v}\|$ 表示。
- 对于环 R 上的两个 n 维向量 $\mathbf{a}, \mathbf{b}$，其内积表示为 $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=0}^{n-1} a_i \cdot b_i \in R$。

2.2 背景知识

2.2.1 格

n 维满秩格 $\mathcal{L}$ 是 $\mathbb{R}^n$ 上的离散加法子群，有性质 $span_{\mathbb{R}}(\mathcal{L}) = \mathbb{R}^n$。格 $\mathcal{L}$ 由一组线性无关的向量 $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_n\}$（称为格基）生成:

$$\mathcal{L} = \mathcal{L}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{Z}^n = \{\sum_{i=1}^{n} z_i \cdot \mathbf{b}_i : z_i \in \mathbb{Z}\}.$$

2.2.2 高斯分布

定义1. （高斯函数）对于任意的实数 $s > 0$，宽度为 s 的一维高斯函数 ρ_s 是 $\mathbb{R} \to \mathbb{R}^+$ 的函数，表示为

$$\rho_s(x) := \exp\left(-\pi\|x\|^2/s^2\right).$$

定义2. （高斯分布）对于任意的实数 $s > 0$，宽度为 s 的一维高斯分布 D_s 是 $\mathbb{R}$ 上的分布，其概率密度函数表示为 $D_s(x) = \rho_s(x)/s$，高斯分布 D_s 的标准差为 $\sigma = s/\sqrt{2\pi}$。

定义3. （离散高斯分布）对于格 $\mathcal{L}$，宽度为 s 的离散高斯分布 $D_{s,\mathcal{L}}$ 是 $\mathcal{L}$ 上的分布，表示为 $D_{s,\mathcal{L}} = \frac{\rho_s(\mathbf{x})}{\rho_s(\mathcal{L})},\ \mathbf{x} \in \mathcal{L}$，其中 $\rho_s(\mathcal{L}) = \sum_{\mathbf{v}\in\mathcal{L}} \rho_s(\mathbf{v})$ 作为归一化因子。

2.2.3 LWE 问题

我们提出的公钥加密方案（PKE）及密钥交换机制（KEM）的安全性依赖于 Learning With Errors （LWE）问题的困难程度，LWE 问题是经典的 Learning Parities with Noise[9] 问题的一般化，在 2005 年由 Regev[31] 首先提出。下面给出 LWE 分布及计算性问题的定义。

定义4. （LWE 分布）n, q是正整数，χ 是 $\mathbb{Z}$ 上的一个分布。给定 $\mathbf{s} \in \mathbb{Z}_q^n$，通过均匀随机的选择 $\mathbf{a} \in \mathbb{Z}_q^n$ 以及从分布 χ 中选择整数错误 $e \in \mathbb{Z}$，LWE 分布 $\mathcal{A}_{s,\chi}$ 输出数对 $(\mathbf{a}, \langle\mathbf{a}, \mathbf{s}\rangle + e \ mod\ q) \in \mathbb{Z}_q^n \times \mathbb{Z}$。

LWE 问题主要分为两种：（1）搜索版本，通过给定一系列 LWE 分布 $\mathcal{A}_{s,\chi}$ 的样本，恢复密钥 $\mathbf{s} \in \mathbb{Z}_q^n$。（2）判定版本，区分从 $\mathbb{Z}_q^n \times \mathbb{Z}$ 中随机均匀采样获得的样本与服从 LWE 分布 $\mathcal{A}_{s,\chi}$ 的样本。对于两种版本的 LWE 问题，$\mathbf{s} \in \mathbb{Z}_q^n$ 的分布一般可认为是均匀的（称为 uniform-secret）或者是属于 $\chi^n\ mod\ q$ 的（称为 normal-form-secret）。

定义5. （搜索版本 LWE 问题）n, m, q 是正整数，χ 是 $\mathbb{Z}$ 上的一个分布。参数为（n, m, q, χ），*uniform-secret*（*normal-form-secret*）的搜索版本 LWE 问题（$SLWE_{n,m,q,\chi}$ 或者 $nf\text{-}SLWE_{n,m,q,\chi}$）为：在 $\mathbf{s} \in \mathbb{Z}_q^n$ 随机均匀选取（或者 $\mathbf{s} \in \chi_q^n$）的条件下，给定 m 个服从 LWE 分布 $\mathcal{A}_{s,\chi}$ 的样本，寻找 $\mathbf{s}$。

定义6. （判定版本 LWE 问题）n, m, q 是正整数，χ 是 $\mathbb{Z}$ 上的一个分布。参数为（n, m, q, χ），*uniform-secret*（*normal-form-secret*）的判定版本 LWE 问题（$DLWE_{n,m,q,\chi}$ 或者 $nf\text{-}DLWE_{n,m,q,\chi}$）为：在 $\mathbf{s} \in \mathbb{Z}_q^n$ 随机均匀选取（或者 $\mathbf{s} \in \chi_q^n$）的条件下，区分 m 个服从 LWE 分布 $\mathcal{A}_{s,\chi}$ 的样本与 m 个服从均匀分布的样本 $U(\mathbb{Z}_q^n \times \mathbb{Z})$。

对于 LWE 问题中的错误分布 χ 一般可取为离散高斯分布，我们将标准差为 α 的离散高斯分布记为 $\Psi_{\alpha\sqrt{2\pi}}$。

3 算法描述

3.1 定义

定义7. （PKE 公钥加密方案）一个公钥加密方案 (PKE) 包括如下三个*PPT* 算法：*KeyGen, Encrypt* 和 *Decrypt*。

- *KeyGen*(1^κ)：密钥生成算法输入安全参数 1^κ, 输出公私钥对（pk, sk）。
- *Encrypt*($pk, \mathbf{m}$)：加密算法输入公钥 pk 和加密消息 $\mathbf{m} \in \mathcal{M}$，输出密文 c。
- *Decrypt*(sk, c)：解密算法输入私钥 sk 和密文 c；输出加密消息 $\mathbf{m}$ 或者 $\perp$（解密失败）。

正确性 如果 ϵ 是一个可忽略数值函数，对所有的 $\mathbf{m} \in \mathcal{M}$, 满足

$$Pr[(pk, sk) \leftarrow KeyGen(1^\kappa) : C \leftarrow Encrypt(pk, \mathbf{m}); Decrypt(sk, c) = \mathbf{m}] > 1 - \epsilon(\lambda)$$

则 PKE 方案正确性得以保证。

定义8. （PKE δ-*correctness[21]*）如果

$$\mathbb{E}\left[\max_{\mathbf{m}\in\mathcal{M}} Pr[\mathrm{PKE.Dec}(sk, c) \neq \mathbf{m} : c \leftarrow \mathrm{PKE.Enc}(pk, \mathbf{m})]\right] \leq \delta,$$

这里的期望取遍 $(pk, sk) \leftarrow$PKE.KeyGen()，则一个消息空间为 $\mathcal{M}$ 的公钥加密方案 PKE 是 δ-*correctness*。

定义9.（KEM 密钥封装方案）一个密钥封装方案（KEM）包括如下三个*PPT*算法：*KeyGen, Encaps* 和 *Decaps*。

- *KeyGen*(1^κ)：密钥生成算法输入安全参数 1^κ，输出公私钥对 (pk, sk)。
- *Encaps*(pk)：封装算法输入公钥 pk，输出密文 c 和封装密钥 K。
- *Decaps*(sk, c)：解封算法输入私钥 sk 和密文 c，输出封装密钥 K。

正确性 如果 ϵ 是一个可忽略数值函数，满足

$$Pr[(pk, sk) \leftarrow KeyGen(1^\kappa) : (c, K) \leftarrow Encaps(pk); Decaps(sk, c) = K] > 1 - \epsilon(\lambda)$$

则 KEM 方案正确性得以保证。

定义10.（PKE 的 *IND-CPA* 安全度）设 PKE=（KeyGen,Encrypt,Decrypt）为公钥加密方案。定义敌手 $\mathcal{A}$ 与挑战者之间的实验：

实验 $IND\text{-}CPA^b_{PKE,\mathcal{A}}(k)$:

- 挑战者运行$(pk, sk) \leftarrow KeyGen(params)$，将 pk 给 $\mathcal{A}$；
- 敌手 $\mathcal{A}$ 输出相同长度消息 $(\mathbf{m}_0, \mathbf{m}_1)$；
- 挑战者计算 $Encrypt(pk, \mathbf{m}_b)$，并将结果给 $\mathcal{A}$；
- $\mathcal{A}$ 输出比特 b'；挑战者将 b' 作为实验输出。

敌手 $\mathcal{A}$ 破解 *IND-CPA* 安全的 PKE 的优势定义为

$$\mathbf{Adv}^{IND-CPA}_{PKE,\mathcal{A}} = |Pr[IND-CPA^1_{PKE,\mathcal{A}}(k) = 1] - Pr[IND-CPA^0_{PKE,\mathcal{A}}(k) = 1]|$$

则如果对任意多项式时间敌手 $\mathcal{A}$ 和任意 k，设 ϵ 是可忽略数值函数，满足 $\mathbf{Adv}^{IND-CPA}_{PKE,\mathcal{A}} \leq \epsilon(k)$，我们称 PKE 为 *IND-CPA* 安全的。

定义11.（KEM 的 *IND-CCA* 安全度）设 KEM=(KeyGen,Encaps,Decaps) 为密钥封装方案。定义敌手 $\mathcal{A}$ 与挑战者之间的实验：

实验 $IND\text{-}CCA^{b}_{KEM,\mathcal{A}}(k)$:

- 挑战者运行 $(pk, sk) \leftarrow KeyGen(params)$，将 pk 给$\mathcal{A}$；
- $\mathcal{A}$ 可访问解封示谕器 $Decaps(sk, \cdot)$；
- 挑战者计算 $(c^*, K_0^*) \leftarrow Encaps(pk)$ 以及 $K_1^* \leftarrow \mathcal{K}$，挑战者将 (c^*, K_b^*) 给 $\mathcal{A}$；
- $\mathcal{A}$ 可以接着访问解封示谕器，但不允许访问密文 c^*；最终 $\mathcal{A}$ 输出比特 b'。挑战者将 b' 作为实验输出。

在量子随机示谕器模型下，挑战者可另外运行随机示谕器。敌手 $\mathcal{A}$ 破解 *IND-CCA* 安全的 KEM 的优势定义为

$$\mathbf{Adv}^{IND-CCA}_{KEM,\mathcal{A}} = |Pr[IND-CCA^{1}_{KEM,\mathcal{A}}(k) = 1] - Pr[IND-CCA^{0}_{KEM,\mathcal{A}}(k) = 1]|$$

则如果对任意多项式时间敌手 $\mathcal{A}$ 和任意 k，设 ϵ 是可忽略数值函数，满足 $\mathbf{Adv}^{IND-CCA}_{KEM,\mathcal{A}} \leq \epsilon(k)$，我们称 KEM 为 *IND-CCA* 安全的。

3.2 采样

错误分布的采样算法宽度直接关系到 LWE 问题的安全性，同时采样算法对公钥加密的效率有重要影响。在现有的基于格困难问题的密码算法设计中，离散错误采样器的采样算法主要可分为三类：1、近似离散高斯分布采样算法；2、二项分布采样算法；3、有界均匀采样算法。前两类采样算法需要一个随机数生成器作为随机源，在一般情况下，随机数生成器产生随机数的效率(比特/秒)稳定，因此，前两类算法输出样本时的随机数利用率(比特/样本)决定了采样算法的效率。第三类算法与前两类算法的不同之处在于其不需要随机源，而需要一个额外的输入 $x \in \mathbb{Z}_q$, 输出则计算 x 乘 $\frac{p}{q}$ 的取整，即 $\lfloor \frac{xp}{q} \rceil$ $(p < q)$，因此对于输入 x，第三类算法的输出是确定的而非随机的，在随机假设下，其输出分布可认为与$[-\frac{q}{2p}, \frac{q}{2p}]$ 上的均匀整数采样相当。

这里引入一种新型的采样方法，我们使用二项分布和有界均匀分布作为采样的底层，通过变量卷积叠加的方式来得到模 q 上的分布，记为 $\chi_{(k_1,k_2)}$。

定义12. 对于服从参数为 (k_1, k_2) 的分布 $\chi_{(k_1,k_2)}$ 的随机变量 X, 有

$$X = \sum_{i=1}^{k_1}(\mathbf{b}_i - \mathbf{b}'_i) + \sum_{i=1}^{k_2}\mathbf{u}_i.$$

其中 $\mathbf{b}_i, \mathbf{b}'_i$ 为服从 $\{0,1\}$ 上的二项分布，即 $Pr[\mathbf{b}_i = 0] = Pr[\mathbf{b}_i = 1] = 1/2$。同时，$\mathbf{u}_i$ 为 $\{-1,0,1\}$ 上的均匀分布。

引理1. 对于服从参数为 (k_1, k_2) 的分布 $\chi_{(k_1,k_2)}$ 的随机变量 X, 有

$$E[X] = 0, D[X] = \frac{k_1}{2} + \frac{2k_2}{3}.$$

证明：鉴于随机变量 X 可以表示为独立随机变量的叠加，即

$$X = \sum_{i=1}^{k_1}(\mathbf{b}_i - \mathbf{b}'_i) + \sum_{i=1}^{k_2}\mathbf{u}_i,$$

其中 $\mathbf{b}_i, \mathbf{b}'_i$ 为服从 $\{0,1\}$ 上的二项分布，$\mathbf{u}_i$ 为 $\{-1,0,1\}$ 上的均匀分布。则 X 的取值范围为 $[-(k_1+k_2),\ k_1+k_2]$, 并且有 $E[X] = 0$,

$$D[X] = 2k_1D[\mathbf{b}_i] + k_2D[\mathbf{u}_i] = \frac{k_1}{2} + \frac{2k_2}{3}.$$

□

可以注意到的是，我们的采样算法为有界均匀与二项的“叠加”，在拥有较高采样效率的同时，通过选取合适的采样参数，我们的算法对比有界均匀分布采样可实现更高的安全性；对比二项分布采可实现更广的参数选择范围；对比近似高斯分布采样可以更准确地估计安全强度。从而在效率、安全性及灵活性等方面均达到优于传统采样算法的目的。

- **安全性方面** 现阶段对于LWE 参数的分析模型主要采用原始攻击和对偶攻击，其中原始攻击关注的是错误分布的长度，对偶攻击使用错误分布的宽

度（即方差）分析，即把错误分布当作理想的离散高斯分布来计算，对偶格中向量所对应的区分优势($\epsilon = e^{-\pi s^2 l^2/q^2}$) 由向量长度和分布宽度估计，因而目前并没有对错误具体分布的安全分析方法. 文章[13]中给出了基于傅里叶变换的区分分析，可以对具体错误分布的区分优势进行分析，并且对实际选取分布与理想高斯分布在对偶攻击下的差异性进行了量化. 结果显示近似取整高斯分布和有界均匀分布，与理想高斯分布估计出的区分复杂度存在一定的差距；而中心二项分布与理想高斯分布具有相同的性质，即区分优势下界与目前的估计ϵ 一致，因而其在现存对偶攻击下的估计是保守的. 在本设计中，从保证安全性出发，我们提出的混合采样同样满足该性质，即ϵ对应对偶格中所有向量的区分优势下界，从而可以获得保守的对偶复杂度估计. 在附录A、B中我们对错误分布的区分分析、混合采样算法给出了简要的描述，详尽的论述和证明参见文献[13]。

- **效率** 混合采样和近似离散高斯采样、中心二项分布、有界均匀的采样效率比对见表Table 14. 在一定的选取规则下，混合采样的效率会高于其他两种采样，参见图Figure 8。

- **参数选取** 根据混合采样的定义可知，有两个参数 k_1, k_2 可以选取，而中心二项分布只有一个参数k，从而该分布相比中心二项分布具有更广的参数选取范围. 分布的宽度会关系到方案的解密失败概率，为了满足期望的方案安全性，分布的宽度、参数n, q的选取是相互限制的，因而更广的参数选取范围会使得方案的整体参数更灵活，提升方案的效率。

3.3 纠错方案

由于加密算法引入了高斯错误，因此我们需要设计纠错方案以确保解密概率。常见的格密码体制中的纠错方案可分为不使用纠错码的方案 (如 NIST 提交方案 Frodo [29]) 和使用纠错码的方案 (如 NIST 提交方案 LAC [26])。Frodo 的纠错方案采用了以下步骤：首先将长度为 $B \cdot \bar{m} \cdot \bar{n}$ 二元消息 $\mathbf{m}$ 按每 B 个比特组合的方式映射为 $\mathbb{Z}_{2^B}$ 上 $\bar{m} \times \bar{n}$ 的整数矩阵，再将此矩阵的每一位乘以 $q/2^B$ 得到一个 $\mathbb{Z}_q^{\bar{m} \times \bar{n}}$ 的整数消息矩阵 $\mathbf{K_m}$。这种纠错方案可得到较小的消息矩阵 $\mathbf{K_m}$ ，从而减少采样个数和加解密的乘法运算次数。但其纠错能力完全由

乘法系数 $q/2^B$ 的大小决定，当 B 固定且高斯错误的宽度增加时，我们必须选取更大的 q 以保证解密成功的概率，这会导致更大的密文大小从而减低传输效率。LAC 的纠错方案中采用了纠错码，即先将长度为 l_e 的二元消息 $\mathbf{m}$ 用二元纠错码映射为长为 n_e 的二元向量，再将向量的每一位乘以 $\bar{q}$ 得到整数消息向量 $\mathbf{k_m}$。这种方案可极大提升纠错能力，从而降低密文大小并提高传输效率。但是其得到的消息向量 $\mathbf{k_m}$ 的长度要大于二元消息 $\mathbf{m}$ 的长度，因此当用于本方案这种无代数结构的 LWE 方案中时，将导致较大的消息矩阵，从而带来更多的采样个数和乘法运算。

因此，在本方案中我们采用二元线性码结合格雷码 (Gray codes) 的纠错方案。一方面，纠错码可提升纠错能力，而格雷码可减小消息矩阵的大小，这使得我们的纠错方案可兼顾上述两种方案的优势。另一方面，格雷码可避免译码时一个错误扩散到多个比特，从而进一步提升本方案的纠错能力。

具体地，一个 $[n_e, l_e, d_e]$ 二元线性码 $\mathcal{C}$ 即 n_e 维线性空间 $\mathbb{Z}_2^{n_e}$ 的一个 l_e 维子空间，其极小距离 d_e 定义为 $\mathcal{C}$ 中非零向量的极小汉明重量，即 $d_e = \min\{\mathrm{w}_H(\mathbf{c}) : \mathbf{c} \in \mathcal{C}, \mathbf{c} \neq \mathbf{0}\}$。我们称 $t_e = \lfloor \frac{d_e-1}{2} \rfloor$ 为 $\mathcal{C}$ 的译码半径。由二元线性码 $\mathcal{C}$ 可定义编码算法 *BCE* 和译码算法 *BCD*，其中

- *BCE(*$\mathbf{u}$*)*: 输入 l_e 维二元向量 $\mathbf{u} \in \{0,1\}^{l_e}$；输出 n_e 维二元向量 $\mathbf{c} \in \{0,1\}^{n_e}$。
- *BCD(*$\mathbf{c}$*)*: 输入 n_e 维二元向量 $\mathbf{c} \in \{0,1\}^{n_e}$；如果存在 $\mathbf{u} \in \{0,1\}^{l_e}$ 使得 $\mathrm{w}_H(BCE(\mathbf{u}) - \mathbf{c}') \leq t_e$，则输出 $\mathbf{u}$，否则译码错误并输出 '$\perp$'。

一个 h-比特格雷码建立了 $\{0,1\}^h$ 到 $\mathbb{Z}_{2^h}$ 之间的一个双射，使得 $\mathbb{Z}_{2^h}$ 中任意两个相邻值所对应的二元向量的汉明距离至多是 1。我们记 *GrayE* 为此 $\{0,1\}^h$ 到 $\mathbb{Z}_{2^h}$ 的映射，记其逆映射为 *GrayD*，即

- *GrayE(*$\mathbf{v}$*)*: 输入一个 h 维二元向量 $\mathbf{v} \in \{0,1\}^h$；输出一个整数 $x \in \mathbb{Z}_{2^h}$。
- *GrayD(*x*)*: 输入 $x \in \mathbb{Z}_{2^h}$；输出一个 h 维二元向量 $\mathbf{v} \in \{0,1\}^h$。对任意 $x, x' \in \mathbb{Z}_{2^h}$，我们有 $|x - x'| = 1 \Rightarrow \mathrm{w}_H(GrayD(x) - GrayD(x')) \leq 1$。

3.3.1 消息编码和译码算法

设 $n_e = h \times \bar{m} \times \bar{n}$ 且 $2^h \mid q$，我们用 *Encode* 函数将任意一个 l_e 比特消息 $\mathbf{m}$ 编码为一个 $\bar{m} \times \bar{n}$ 的模 q 矩阵 $\mathbf{K}$。此函数分为三个部分，第一步使用 $[n_e, l_e, d_e]$ 二元线性码的编码算法 *BCE* 将 $\mathbf{m}$ 变换为一个长度为 n_e 的二元向量 $\mathbf{u}$；第二步利用格雷码将 $\mathbf{u}$ 映射为一个 $\bar{m} \times \bar{n}$ 的模 2^h 矩阵 $\mathbf{W}$，即

$$\mathbf{W}_{i,j} = \mathit{GrayE}(\mathbf{u}^{(i,j)}),\ \forall\ 0 \le i < \bar{m},\ 0 \le j < \bar{n},$$

这里 $\mathbf{u}^{(i,j)} = (\mathbf{u}_{(i+\bar{m}j)h}, \mathbf{u}_{(i+\bar{m}j)h+1}, \cdots, \mathbf{u}_{(i+\bar{m}j)h+h-1})$。第三步将矩阵 $\mathbf{W}$ 的每个位置乘以 $q/2^h$，得到矩阵 $\mathbf{K}$。

反之，*Decode* 函数将任意一个 $\bar{m} \times \bar{n}$ 的模 q 矩阵 $\mathbf{K}$ 译码为 l_e 比特消息 $\mathbf{m}$。容易验证，对任意消息 $\mathbf{m} \in \{0,1\}^{l_e}$，有 $\mathit{Decode}(\mathit{Encode}(\mathbf{m})) = \mathbf{m}$。实际上，此方案的纠错能力可由如下引理刻画。

引理2. 对任意矩阵 $\mathbf{E} \in \mathbb{Z}_q^{\bar{m} \times \bar{n}}$ 满足

$$\sum_{\substack{0 \le i < \bar{m} \\ 0 \le j < \bar{n}}} \left| \lfloor \frac{2^h}{q} \cdot \mathbf{E}_{i,j} \rceil \right| \le t, \tag{1}$$

有 $\mathit{Decode}(\mathit{Encode}(\mathbf{m}) + \mathbf{E}) = \mathbf{m}$。

证明. 令 $\mathbf{u} = \mathit{BCE}(\mathbf{m}), \mathbf{W} = \mathit{GrayE}(\mathbf{u}), \mathbf{W}' = \lfloor \frac{2^h}{q} \cdot (\mathit{Encode}(\mathbf{m}) + \mathbf{E}) \rceil, \mathbf{u}' = \mathit{GrayD}(\mathbf{W}')$。由

$$\begin{aligned} \mathbf{W}' &= \lfloor 2^h/q \cdot (\mathit{Encode}(\mathbf{m}) + \mathbf{E}) \rceil \\ &= 2^h/q \cdot \mathit{Encode}(\mathbf{m}) + \lfloor 2^h/q \cdot \mathbf{E} \rceil \\ &= \mathbf{W} + \lfloor 2^h/q \cdot \mathbf{E} \rceil \end{aligned}$$

可得 $\sum \psi_{2^h}(|\mathbf{W}_{i,j} - \mathbf{W}'_{i,j}|) \le t$，这里 $\psi_{2^h}(x) = \min\{x, 2^h - x\}, \forall\ 0 \le x < 2^h$。因此由格雷映射的性质可知 $\mathrm{w}_H(\mathbf{u} - \mathbf{u}') \le t$，再由纠错码的性质可得 $\mathit{BCD}(\mathbf{u}') = \mathbf{m}$。 □

3.3.2 纠错方案对比

针对 LWE 加密中的纠错问题，SCloud 采取了先对所有明文比特使用长纠错码整体引入冗余，再利用格雷码减小消息矩阵规模的方案。由于长码的纠错能力更强，且格雷码具有控制错误比特扩散的特性，这使得 SCloud 中的纠错方案能够提供更高的传输及运算效率。同时，长纠错码结合格雷码的方式也使得 SCloud 纠错方案的参数选取更加灵活。

下面我们将 SCloud 中的纠错方案与 Frodo 纠错方案（仅用简单映射控制矩阵规模，且不使用长纠错码）及 LAC 中的纠错方案 (仅使用长纠错码，而不控制消息矩阵规模) 做一个对比。根据第 6 章的分析，SCloudPKE 方案中的密文长度为 $(\bar{m}n + \bar{m}\bar{n})\log q$，加密运算需进行 $2\bar{m}n + \bar{m}\bar{n}$ 个采样及 $n^2\bar{m} + \bar{m}\bar{n}n$ 次整数乘法及同样次数的整数加法，解密运算需进行 $\bar{m}\bar{n}n$ 次整数乘法和同样次数整数加法。

在 f640 参数下，使用我们的纠错方案可得消息矩阵的大小为 $\bar{m} \times \bar{n} = 9 \times 11$，$q$ 的大小为 2^{14}，因此密文长度为 82026 bit ，加密运算需进行 11619 个采样及 3749760 次整数乘法/加法，解密运算需进行 63360 次整数乘法/加法。使用 Frodo 中的纠错方案，若要求解密失败概率小于 2^{-128} 且限制 $q \leq 2^{14}$ ，则消息矩阵的大小至少为 $\bar{m} \times \bar{n} = 16 \times 16$，$q$ 为 2^{13}，此时密文长度为 136448 bit ，加密运算需进行 20736 个采样及 6717440 次整数乘法/加法，解密运算需进行 163840 次整数乘法/加法。若使用 LAC 中的纠错方案，要求解密失败概率小于 2^{-128} 且 $q \leq 2^{14}$，则 $(\bar{m}, \bar{n}, q) = (16, 17, 2^12)$ 或 $(16, 16, 2^13)$ [1]，两组参数对应的密文长度分别为 126144 $bit, 136448$ bit ，加密运算分别需 $20752, 20736$ 个采样及 $6727680, 6717440$ 次整数乘法/加法，解密运算分别需 $174080, 163840$ 次整数乘法/加法。

3.4 IND-CPA PKE

此处我们描述公钥加密方案 SCloudCPAPKE，其可实现 IND-CPA 安全度。方案描述见 Figure 1。方案包括算法（SCouldPKE.KeyGen, SCloudCPAPKE.Enc, SCloudCPAPKE.Dec），算法分别在 Algorithm 1，Algorithm

[1]此时对应于使用 $[n_e = l_e, l_e, d_e = 1]$ 的平凡纠错码。

2，Algorithm 3。

Initiator		Responder
KeyGen()$\rightarrow (pk, sk)$:		
$\mathbf{seed_A} \leftarrow \{0,1\}^{\kappa}$		
$\mathbf{A} = Gen(\mathbf{seed_A}) \in \mathbb{Z}_q^{n\times n}$		
$\mathbf{S}, \mathbf{E} \leftarrow \chi^{n\times\bar{n}}$		
$\mathbf{B} = \mathbf{AS} + \mathbf{E}$		Enc($\mathbf{m} \in \mathbb{Z}^{l_e}$,$pk$)$\rightarrow c$:
$sk \leftarrow \mathbf{S}$; $pk \leftarrow (\mathbf{seed_A}, \mathbf{B})$	$\xrightarrow{pk}$	$\mathbf{A} = Gen(\mathbf{seed_A})$
		$\mathbf{S}', \mathbf{E}' \leftarrow \chi^{\bar{m}\times n}$;$\mathbf{E}'' \leftarrow \chi^{\bar{m}\times\bar{n}}$
		$\mathbf{B}' = \mathbf{S}'\mathbf{A} + \mathbf{E}'$
		$\mathbf{V} = \mathbf{S}'\mathbf{B} + \mathbf{E}'' + Encode(\mathbf{m}) \in \mathbb{Z}^{\bar{m}\times\bar{n}}$
Dec(c, sk)$\rightarrow \mathbf{m}$:	$\xleftarrow{c}$	$c \leftarrow (\mathbf{B}', \mathbf{V})$
$\mathbf{D} = \mathbf{V} - \mathbf{B}'\mathbf{S}$		
$\mathbf{m} = Decode(\mathbf{D})$		

Figure 1: IND-CPA PKE 方案 SCloudCPAPKE

Algorithm 1 SCloudCPAPKE.KeyGen

Input: None

Output: (pk, sk)

1: 选择随机种子 $\mathbf{seed_A}$
2: 使用随机种子生成随机矩阵 $\mathbf{A} \leftarrow Gen(\mathbf{seed_A}) \in \mathbb{Z}_q^{n\times n}$
3: 生成满足高斯分布的矩阵 $\mathbf{S}, \mathbf{E} \leftarrow \chi^{n\times\bar{n}}$
4: 计算 $\mathbf{B} = \mathbf{AS} + \mathbf{E}$
5: 返回公钥 $pk \leftarrow (\mathbf{seed_A}, \mathbf{B})$, 以及私钥 $sk \leftarrow \mathbf{S}$

3.5 PKE 正确性

解密算法 Dec 计算：

$$\begin{aligned}\mathbf{V} - \mathbf{B}'\mathbf{S} &= \mathbf{S}'\mathbf{B} + \mathbf{E}'' + Encode(\mathbf{m}) - (\mathbf{S}'\mathbf{AS} + \mathbf{E}'\mathbf{S}) \\ &= \mathbf{S}'\mathbf{AS} + \mathbf{S}'\mathbf{E} + \mathbf{E}'' + Encode(\mathbf{m}) - \mathbf{S}'\mathbf{AS} - \mathbf{E}'\mathbf{S} \\ &= \mathbf{S}'\mathbf{E} + \mathbf{E}'' - \mathbf{E}'\mathbf{S} + Encode(\mathbf{m})\end{aligned}$$

Algorithm 2 SCloudCPAPKE.Enc

Input: 消息 $\mathbf{m} \in \mathbb{Z}^{l_e}$, 公钥 $pk = (\mathbf{seed_A}, \mathbf{B})$
Output: 密文 c
1: 使用随机种子生成随机矩阵 $\mathbf{A} \leftarrow Gen(\mathbf{seed_A}) \in \mathbb{Z}_q^{n \times n}$
2: 生成满足高斯分布的矩阵 $\mathbf{S}', \mathbf{E}' \leftarrow \chi^{\bar{m} \times n}$,$\mathbf{E}'' \leftarrow \chi^{\bar{m} \times \bar{n}}$
3: 计算 $\mathbf{B}' = \mathbf{S}'\mathbf{A} + \mathbf{E}'$ 以及 $\mathbf{V} = \mathbf{S}'\mathbf{B} + \mathbf{E}'' + Encode(\mathbf{m})$
4: 返回密文 $c \leftarrow (\mathbf{B}', \mathbf{V})$

Algorithm 3 SCloudCPAPKE.Dec

Input: 密文 $c = (\mathbf{B}', \mathbf{V}) \in \mathbb{Z}_q^{\bar{m} \times n} \times \mathbb{Z}_q^{\bar{m} \times \bar{n}}$, 私钥 $sk = \mathbf{S} \in \mathbb{Z}_q^{n \times \bar{n}}$
Output: 解密消息 $\mathbf{m}$
1: 计算 $\mathbf{D} = \mathbf{V} - \mathbf{B}'\mathbf{S}$
2: 返回消息 $\mathbf{m} = Decode(\mathbf{D})$

记 $\mathbf{E}''' = \mathbf{S}'\mathbf{E} + \mathbf{E}'' - \mathbf{E}'\mathbf{S}$，则由引理 2 可知解密成功当且仅当 $\mathbf{E}'''$ 满足式 (1) 条件。

矩阵 $\mathbf{E}'''$ 的每一个位置都可看作 $2n+1$ 个独立随机变量之和，其中 $2n$ 个是两个由 χ 生成的独立高斯变量的乘积，另一个是由 χ 生成的高斯变量。将此 $2n+1$ 个变量之和记为变量 χ'，则解密失败概率可表示为

$$p_{error} = 1 - \sum_{\sum |\lfloor (2^h/q) \cdot \mathbf{E}'''_{i,j} \rceil| \leq t} \prod_{i,j} \chi'(\mathbf{E}'''_{i,j}).$$

3.6 将IND-CPA PKE转换为IND-CCA KEM

使用 Fujisaki-Okamoto 转换[17] 可以在经典随机示谕器模型下使用一个 one-way-secure 安全的公钥加密方案构造出一个 IND-CCA2 安全的公钥加密方案。Targhi 和 Unruh[34] 给出了一个 Fujisaki-Okamoto 变换的变形，然后给出了量子敌手下基于量子随机示谕器的 IND-CCA2 安全性。Hofheinz, Hővelmanns 和 Kiltz[21] 给出了一种在量子随机示谕器下的变形，在 NIST 提交方案 Frodo[29] 中作者使用 Hofheinz, Hővelmanns 和 Kiltz 的技术，将 IND-CPA 公钥加密方案和两个哈希函数构造出 IND-CCA 安全的密钥封装方案，我们将使用它们的构造作为模板。此处我们描述密钥封装 SCLoudKEM，其可实现 IND-CCA 安全性。方案见 Figure 2。方案包括算法（SCloudKEM.KeyGen，

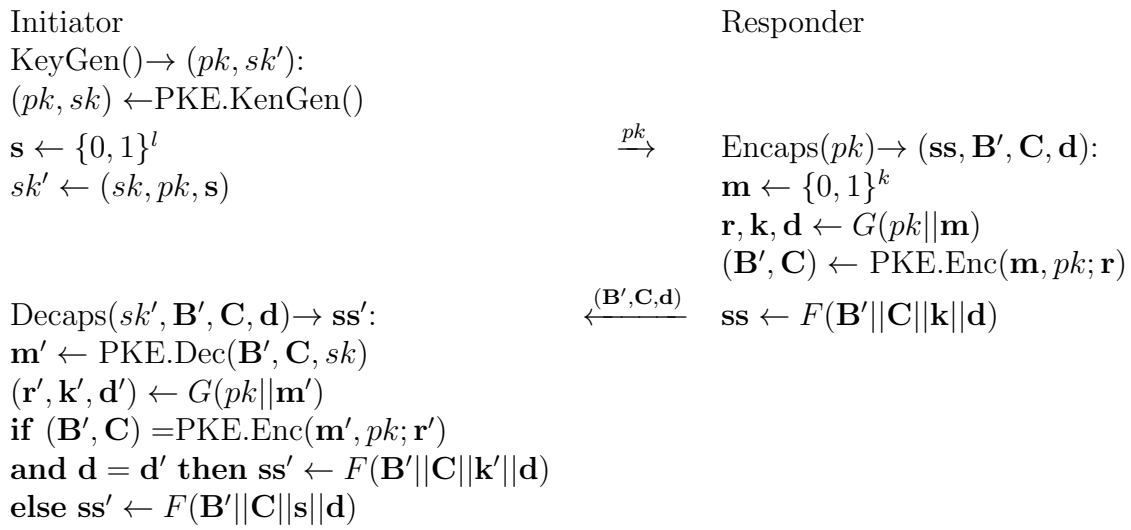

Figure 2: IND-CCA KEM 方案 SCloudKEM.

SCloudKEM.Encaps，SCloudKEM.Decaps），其在Algorithm 4，Algorithm 5，Algorithm 6。

Algorithm 4 SCloudKEM.KeyGen

Input: None

Output: 公私钥 (pk, sk')

1: 生成随机种子 $\mathbf{s}||\mathbf{seed_E}||\mathbf{seed_A}$.
2: 使用随机种子 $\mathbf{seed_A}$ 生成随机矩阵 $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$: $\mathbf{A} \leftarrow Gen(\mathbf{seed_A})$
3: 使用随机种子 $\mathbf{seed_E}$ 生成错误矩阵 $\mathbf{S}, \mathbf{E} \leftarrow \chi^{n \times \bar{n}}$
4: 计算 $\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$
5: 返回公钥 $pk \leftarrow (\mathbf{seed_A}, \mathbf{B})$, 密钥 $sk' \leftarrow (\mathbf{s}, \mathbf{seed_A}, \mathbf{B}, \mathbf{S})$

3.7 IND-CCA KEM的正确性

IND-CCA SCloudKEM 的解密失败概率与前面 IND-CPA SCloudCPAPKE 的解密失败概率相同。

Algorithm 5 SCloudKEM.Encaps()

Input: $pk = (\mathbf{seed_A}, \mathbf{B})$
Output: 密文 $(\mathbf{B'}, \mathbf{C}, \mathbf{d})$

选择随机消息 $\mathbf{m}$
生成随机种子 $\mathbf{r}, \mathbf{k}, \mathbf{d} \leftarrow G(\mathbf{seed_A}||\mathbf{B}||\mathbf{m})$
使用随机种子 $\mathbf{r}$ 生成错误矩阵 $\mathbf{S'}, \mathbf{E} \leftarrow \chi^{\bar{m}\times n}$,$\mathbf{E''} \leftarrow \chi^{\bar{m}\times\bar{n}}$
生成矩阵 $\mathbf{A} \leftarrow Gen(\mathbf{seed_A})$
计算 $\mathbf{B'} \leftarrow \mathbf{S'A} + \mathbf{E}$
计算 $\mathbf{V} \leftarrow \mathbf{S'B} + \mathbf{E''}$
计算 $\mathbf{C} \leftarrow \mathbf{V} + Encode(\mathbf{m})$
计算 $\mathbf{ss} \leftarrow F(\mathbf{B'}||\mathbf{C}||\mathbf{k}||\mathbf{d})$
返回密文 $\mathbf{B'}, \mathbf{C}, \mathbf{d}$ 和协商密钥 $\mathbf{ss}$

Algorithm 6 SCloudKEM.Decaps()

Input: 密文 $\mathbf{B'}, \mathbf{C}, \mathbf{d}$, 密钥 $sk' \leftarrow (\mathbf{s}, \mathbf{seed_A}, \mathbf{B}, \mathbf{S})$
Output: 协商密钥 $\mathbf{ss}$

计算 $\mathbf{D} = \mathbf{C} - \mathbf{B'S}$
计算 $\mathbf{m'} \leftarrow Decode(\mathbf{D})$
生成随机种子 $\mathbf{r'}, \mathbf{k'}, \mathbf{d'} \leftarrow G(\mathbf{seed_A}||\mathbf{B}||\mathbf{m'})$
使用随机种子 $\mathbf{r'}$ 生成错误矩阵 $\mathbf{S'}, \mathbf{E'} \leftarrow \chi^{\bar{m}\times n}$,$\mathbf{E''} \leftarrow \chi^{\bar{m}\times\bar{n}}$
生成矩阵 $\mathbf{A} \leftarrow Gen(\mathbf{seed_A})$
计算 $\mathbf{B''} \leftarrow \mathbf{S'A} + \mathbf{E'}$
计算 $\mathbf{V} \leftarrow \mathbf{S'B} + \mathbf{E''}$
计算 $\mathbf{C'} \leftarrow \mathbf{V} + Encode(\mathbf{m'})$
if $(\mathbf{B'}||\mathbf{C} = \mathbf{B''}||\mathbf{C'})$ and $(\mathbf{d} = \mathbf{d'})$ **then**
  return $\mathbf{ss} \leftarrow F(\mathbf{B'}||\mathbf{C}||\mathbf{k'}||\mathbf{d})$
else
  return $\mathbf{ss} \leftarrow F(\mathbf{B'}||\mathbf{C}||\mathbf{s}||\mathbf{d})$
end if

Initiator		Responder
KeyGen()$\rightarrow (pk, sk')$:		
$(pk, sk') \leftarrow$KEM.KenGen()	$\xrightarrow{pk}$	Enc($\mathbf{ms} \in \mathbb{Z}^{l_m}$,$pk$)$\rightarrow (\mathbf{B}', \mathbf{C}, \mathbf{d}, \mathbf{K})$: $(\mathbf{ss}, \mathbf{B}', \mathbf{C}, \mathbf{d}) \leftarrow$KEM.Encaps($pk$)
Decaps($sk', \mathbf{B}', \mathbf{C}, \mathbf{d}, \mathbf{K}$)$\rightarrow \mathbf{ms}'$: $\mathbf{ss}' \leftarrow$KEM.Decaps($\mathbf{B}', \mathbf{C}, \mathbf{d}, sk'$) $\mathbf{ms} \leftarrow \mathrm{DEM}^{-1}(\mathbf{ss}', c2)$	$\xleftarrow{(\mathbf{B}', \mathbf{C}, \mathbf{d}, c2)}$	$c2 = \mathrm{DEM}(\mathbf{ss}, \mathbf{ms})$

Figure 3: IND-CCA PKE 方案 SCloudCCAPKE

3.8 转换为IND-CCA PKE

通过使用 IND-CCA SCloudKEM 与一个数据封装机制 DEM，可以构造一个 IND-CCA 安全的加密方案 SCloudCCAPKE，这种转换方式最早被 Cramer 和 Shoup 提出[16]，并被广泛应用在 IND-CCA 的加密构造中。在 SCloudCCAPKE 的加密算法中，首先使用 SCloudKEM 封装出一个密钥 **ss**，然后该密钥被用到 DEM 中加密明文 **ms**。在解密算法中，首先使用 SCloudKEM 解封装出密钥 **ss**, 然后使用 DEM 解密传输密文 $c2$ 得到明文 **ms**。方案见 Figure 3，包括算法（SCloudCCAPKE.KeyGen，SCloudCCAPKE.Enc，SCloudCCAPKE.Dec），其在Algorithm 7，Algorithm 8，Algorithm 9。

Algorithm 7 SCloudCCAPKE.KeyGen

Input: None
Output: (pk, sk')
1: 使用 SCloudKEM.KeyGen() 生成公私钥对(pk, sk')
2: 返回公钥 pk, 以及私钥 sk'

Algorithm 8 SCloudCCAPKE.Enc

Input: 消息 $\mathbf{ms} \in \mathbb{Z}^{l_m}$, 公钥 pk
Output: 密文 c
1: 使用 SCloudKEM.Encaps(pk) 封装出 $(\mathbf{ss}, \mathbf{B}', \mathbf{C}, \mathbf{d})$
2: 使用 DEM 封装加密消息 $c2 = \mathrm{DEM}(\mathbf{ss}, \mathbf{ms})$
3: 返回密文 $c \leftarrow (\mathbf{B}', \mathbf{C}, \mathbf{d}, c2)$

Algorithm 9 SCloudCCAPKE.Dec

Input: 密文 $c = (\mathbf{B}', \mathbf{C}, \mathbf{d}, c2)$, 私钥 sk'
Output: 解密消息 **ms**
1: 使用 SCloudKEM.Decaps$(\mathbf{B}', \mathbf{C}, \mathbf{d}, sk')$ 解封出 $\mathbf{ss}'$
2: 使用 DEM$^{-1}(\mathbf{ss}', c2)$ 解密出 **ms**
3: 返回消息 **ms**

3.9 加密原型

方案我们用到了几个加密原型。如下为他们的安全要求：

- Gen: 为一个公共随机函数被用作生成随机矩阵 $\mathbf{A}$. 我们的方案 Gen 算法使用 SM3 哈希算法。
- G: 我们需要 G 为一个伪随机数生成器来生成 KEM 中所需要的随机种子。G 算法使用 SM3 哈希算法。
- F: 我们需要 F 为一个伪随机数生成器来生成 KEM 中最终协商的密钥。F 算法使用 SM3 哈希算法。
- H: PKE 与 KEM 需要一个为随机数生成器生成 $\mathbf{seed_A}$。我们的方案中使用 SM3 哈希算法。
- DEM: 在DEM中我们使用的是AES-GCM加密算法。

4 参数选取

下面我们给出一组可以高效实现的SCloudCPAPKE参数，高斯参数选取的为较小的参数，并且消息比特是安全强度倍数以抵抗Grover搜索攻击（见Table 3），所以有更快速的算法实现，虽然缺乏安全归约证明，但其在有限样本下仍满足抵抗原始攻击和对偶攻击的安全强度要求。参数f640可满足算法竞赛中要求2，3，4。我们在附录中给出适应其他情况的参数选取，包括消息比特同安全等级吻合的高效参数（Table 20），量子安全参数（Table 19）以及消息比特是安全强度倍数的量子安全参数（Table 15）。从这些不同种类的参数选择上可以看出，我们的算法通过适当选取参数能够适应不同情景。

Table 1: SCloudCPAPKE高效的参数选取

	f640	f896	f1216
安全等级（bit）	128	192	256
n	640	896	1216
q	2^{14}	2^{14}	2^{15}
$(\bar{m}, \bar{n})$	$(9, 11)$	$(12, 14)$	$(13, 15)$
σ	2.12	2.12	2.12
消息m比特数	256	384	512
(k_1, k_2) in χ_{k_1,k_2}	(5,3)	(5,3)	(5,3)
解密失败概率	2^{-135}	2^{-206}	2^{-295}
$[n_e, l_e, d_e]$ 线性码	[292,256,9]	[474,384,21]	[552,512,9]
h-比特格雷码	3	3	3

Table 2: SCloudCCAKEM高效的参数选取

	f640	f896	f1216
安全等级（bit）	128	192	256
n	640	896	1216
q	2^{14}	2^{14}	2^{15}
$(\bar{m}, \bar{n})$	$(9, 11)$	$(12, 14)$	$(13, 15)$
σ	2.12	2.12	2.12
协商ss比特数	256	384	512
(k_1, k_2) in χ_{k_1,k_2}	(5,3)	(5,3)	(5,3)
解密失败概率	2^{-135}	2^{-206}	2^{-295}
$[n_e, l_e, d_e]$ 线性码	[292,256,9]	[474,384,21]	[552,512,9]
h-比特格雷码	3	3	3

5 安全性分析

5.1 安全归约

5.1.1 SCloudCPAPKE 的 IND-CPA 安全性

定理1. 令 $n, q, \bar{m}, \bar{n}$为正整数，χ 为 $\mathbb{Z}$ 上的概率分布；假设矩阵 $\mathbf{A}$ 为均匀矩阵，对任意针对 SCloudCPAPKE 的 *IND-CPA* 安全性攻击的量子敌手 $\mathcal{A}$，存在

Table 3: SCloudCCAPKE高效的参数选取

	f640	f896	f1216
安全等级（bit）	128	192	256
n	640	896	1216
q	2^{14}	2^{14}	2^{15}
$(\bar{m}, \bar{n})$	$(9, 11)$	$(12, 14)$	$(13, 15)$
σ	2.12	2.12	2.12
消息m比特数	256	384	512
(k_1, k_2) in χ_{k_1,k_2}	(5,3)	(5,3)	(5,3)
解密失败概率	2^{-135}	2^{-206}	2^{-295}
$[n_e, l_e, d_e]$ 线性码	[292,256,9]	[474,384,21]	[552,512,9]
h-比特格雷码	3	3	3

攻击一般形式 LWE 的判定问题的量子算法 $\mathcal{B}_1$ 和 $\mathcal{B}_2$ 使得

$$\mathbf{Adv}^{ind-cpa}_{SCloudCPAPKE}(\mathcal{A}) \leq \bar{n} \cdot \mathbf{Adv}^{nf-dlew}_{n,q,\chi}(\mathcal{B}_1) + \bar{m} \cdot \mathbf{Adv}^{nf-dlew}_{n,n+\bar{n},q\chi}(\mathcal{B}_2).$$

这里算法 $\mathcal{B}_1$ 和 $\mathcal{B}_2$ 的运行时间逼近算法 $\mathcal{A}$ 的时间。

定理1的证明与[25]中的定理1类似。

5.1.2 SCloudKEM 的 IND-CCA 安全性

定理2. 令 PKE 为一个公钥加密方案，其包含算法(KeyGen, Enc, Dec)，消息空间为 $\mathcal{M}$，且为 $\delta - correct$。令 G 和 F 为独立的随机示谕器。对任意针对 SCloudKEM 的 *IND-CCA* 安全性攻击的经典敌手 $\mathcal{A}$，其访问 q_G 次 G，q_F 次 F，则存在针对 PKE 方案的 *IND-CPA* 安全性的经典算法 $\mathcal{B}$ 使得

$$\mathbf{Adv}^{ind-cca}_{SCloudKEM}(\mathcal{A}) \leq \frac{4 \cdot q_{RO} + 1}{|\mathcal{M}|} + q_{RO} \cdot \delta + 3 \cdot \mathbf{Adv}^{ind-cpa}_{PKE}(\mathcal{B})$$

这里 $q_{RO} = q_G + q_F$，这里 $\mathcal{B}$ 的运行时间约为 $\mathcal{A}$ 的时间。

定理3. 令 PKE 为一个公钥加密方案，其包含算法 (KeyGen, Enc, Dec)，消息空间为 $\mathcal{M}$，且为$\delta - correct$。令 G 和 F 为独立的随机示谕器。对任意针对 SCloudKEM 的 *IND-CCA* 安全性攻击的量子敌手 $\mathcal{A}$，其访问 q_G 次 G，q_F

次 F，则存在针对 PKE 方案的 *IND-CPA* 安全性的量子算法 $\mathcal{B}$ 使得

$$\mathbf{Adv}^{ind-cca}_{SCloudKEM}(\mathcal{A}) \leq 9 \cdot q_{RO} \cdot \sqrt{q^2_{RO} \cdot \delta + q_{RO} \cdot \sqrt{\mathbf{Adv}^{ind-cpa}_{PKE}(\mathcal{B}) + \frac{1}{|\mathcal{M}|}}}$$

这里 $q_{RO} = q_G + q_F$，这里 $\mathcal{B}$ 的运行时间约为 $\mathcal{A}$ 的时间。

定理2与定理3的证明类似于[29]中定理5.1与定理5.2。

5.1.3 SCloudCCAPKE 的 IND-CCA 安全性

注意到我们方案由IND-CCA KEM转换到IND-CCA PKE的方式为通用方式，SCloudCCAPKE的IND-CCA安全性可以参照其他使用此转换的方案，如Round5[6]。

5.2 攻击分析

在这一节，我们来解释如何估计方案的安全强度并给出当下攻击 LWE 方案的几种常用方法。我们记 m_{samp} 为攻击者能获得的 LWE 样本数。根据获得样本数量的多少，存在着不同的攻击方法：获得小数目的样本（$m_{samp} \approx n$）时可采用两种 BKZ 类型攻击方法－原始攻击与对偶攻击；获得样本数量为（$m_{samp} = O(n^{2t+1})$）时可采用代数攻击方法[5]；获得指数级的样本时（$m_{samp} = 2^{O(n)}$）可以采用 BKW 攻击[22]。接下来，我们将介绍前两种攻击方法，代数攻击与BKW攻击参见附录C，D。

5.2.1 BKZ 类型攻击

BKZ[12] 算法是 LLL 算法的改进版本，将 LLL 算法中在 2 维空间内寻找短向量扩展为在 k 维空间中寻找短向量，从而获得更优的 Hermite 因子 $\delta^n = \|\mathbf{b}_1\|/vol(\mathcal{L})^{1/n}$。相对应的，LLL 算法可以被严格证明是多项式时间终止的算法，而 BKZ 算法则需要进行指数多次运算。[20] 中提到，在进

行 $round = n^2 \log n/k^2$ 轮运算后的基接近于最终输出，因此 BKZ 算法一般都是进行多项式轮输出后终止。

下面，我们需要估计 BKZ 算法的时间复杂度。分块大小为 b 的 BKZ 算法需要对 b 维 SVP 示谕器进行多项式次数的访问。考虑到未来可能的算法改进，我们只对一次 b 维 SVP 示谕器的时间进行计数，我们称之为核心 SVP 问题的困难度。

然而，准确估计 b 维 SVP 的时间复杂度也是很困难的，因为已经被数次优化过的裁剪枚举策略没有一个准确的表达式[12]。我们只知道枚举是超指数级的，而筛选算法是指数级的：$2^{cb+o(b)}$ （对常数参数 c 已经进行过广泛的研究）。当我们确定了分块大小的范围，可以用筛法的复杂度对枚举复杂度进行下界估计，则枚举算法的时间复杂度至少为 2^{cb}。

已知的指数常数 c 的估计有：对于经典算法为 $c_C = \log_2 \sqrt{3/2} \approx 0.292$，此参数由[8] 得出。对量子算法来说，此常数为 $c_Q = \log_2 \sqrt{13/9} \approx 0.265$。因为所有的筛选算法的变形都需要建立 $(\sqrt{4/3})^b$ 大小的向量列表，所以可以视 $c_P = \log_2 \sqrt{4/3} \approx 0.2075$ 为最好的筛选算法时间复杂度的下界。

上述估计相比于实际应用[28] 来说是很保守的，实际实验中当 b 的范围在 $60 - 80$ 之间时,实际复杂度为大约 $2^{0.405b+11}$，而经典估计的时间复杂度为 $2^{0.292b}$。

当然，我们也要考虑基于当前技术的改进攻击。至少，我们可以假设 $2^{0.292b+o(b)}$ 是找到 SVP 问题解的理想时间复杂度，因为在现有研究中使用[8] 中的技术来解决一般最近邻问题[4] 时能达到此下界。目前，其他算法的时间复杂度提升多是通过在此算法上进行微调获得的，所以我们可以认定如果没有新的思路，此时间复杂度并不会有较大改进。

5.2.2 原始攻击

原始攻击是对 LWE 问题的实例构造一个有唯一最短向量的格，然后用 BKZ 求解格的最短向量。结合上述核心 SVP 困难度计算，我们需要估计用 BKZ 算法找到正确解使用的分块大小的下界。

具体来说，给定 LWE 实例矩阵 $(\mathbf{A}, \mathbf{b} = \mathbf{As}+\mathbf{e})$，构造维度为 $d = m+n+1$ 的格 $\Lambda = \{\mathbf{x} \in \mathbb{Z}^{m+n+1} : (\mathbf{A}|\mathbf{I}_m| - \mathbf{b})\mathbf{x} = \mathbf{0}\ mod\ q\}$，此格的体积为 q^m，并且有唯一最短向量解 $\mathbf{v} = (\mathbf{s}, \mathbf{e}, 1)$，最短向量的模长为 $\lambda \approx \sigma\sqrt{n+m}$。用到的样本数 $m \in [0, 2n]$，具体数值可根据实际需求通过数值优化的方法确定。

使用典型 BKZ 模型并根据等比级数假设（GSA），可将 $\{\|\mathbf{b}_i^*\|\}_{i=0}^{d-1}$ 看作等比数列：$\|\mathbf{b}_i^*\| = \delta^{d-2i-1} \cdot vol(\mathcal{L})^{1/d}$。若最短向量 $\mathbf{v}$ 在最后 b 维空间中投影长度比 $\|\mathbf{b}_{d-b}^*\|$ 短，则可以通过 BKZ 算法求得。因此，可以得出结论：原始攻击成功当且仅当 $\sigma\sqrt{b} \leq \delta^{2b-d-1} \cdot q^{m/d}$，这里参数 $\delta = ((\pi b)^{1/b}\frac{b}{2\pi e})^{1/(2(b-1))}$。使用此公式我们可以获得一个对分块大小 b 的下界估计,再结合核心 SVP 的困难度可以给出 BKZ 算法复杂度的一个下界估计。

5.2.3 对偶攻击

对偶攻击是一类区分攻击，它的基本思路是：首先，利用 BKZ 算法寻找对偶格 $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2) \in \Lambda' = \{(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}^m \times \mathbb{Z}^n : \mathbf{A}^t\mathbf{x} = \mathbf{y}\ mod\ q\}$ 中的短向量。然后，使用短向量 $\mathbf{w}$ 作为 LWE 的区分器 $z := \mathbf{w}_1^t \cdot \mathbf{b} = \mathbf{w}_1^t\mathbf{As} + \mathbf{w}_1^t\mathbf{e} = \mathbf{w}_2^t\mathbf{s} + \mathbf{w}_1^t\mathbf{e}\ mod\ q$。注意对偶格 Λ' 的维数 $d = m+n$，体积为 q^n。执行分块大小为 b 的 BKZ 算法，能输出长度为 $l = \|\mathbf{b}_0\| = \delta^{d-1}q^{n/d}$ 的短向量。

在找到长度为 l 向量的前提下，攻击者可以计算 $z = \mathbf{w}_2^t\mathbf{s} + \mathbf{w}_1^t\mathbf{e}\ mod\ q$。若 $(\mathbf{A}, \mathbf{b})$ 是一个 LWE 样本，z 的分布一个标准差为 $l\sigma$ 为的高斯分布，否则，z 的分布是模 q 上的均匀分布。这两个分布方差的最大距离为 $\varepsilon = 4\exp(-2\pi^2\tau^2),\ \tau = l\sigma/q$，所以给定长度为 l 向量，攻击者可以以 ε 优势区分 LWE 样本与随机样本。

小优势 ε 在攻击密钥交换方案时并没有任何意义，因为最终协商的密钥被作为对称加密方案密钥，任何小于 1/2 的优势都不会明显减少对称加密方案的穷搜空间。

所以我们需要大约 $1/\varepsilon^2$ 个短向量来放大成功概率，筛选算法可以提供 $2^{0.2075b}$ 个短向量，理想条件下攻击者重复 $R = max(1, 1/(2^{2075b}\varepsilon^2))$ 次即可获得足够向量。

对偶攻击的总时间复杂度为 $R \cdot 2^{cb}$，我们可以选取一个最优的 b 使得时间复杂度尽可能低。

在Table Table 4 给出高效参数的原始攻击与对偶攻击的攻击强度。表中参数 prime 代表原始攻击，dual 代表对偶攻击，classical 代表已知经典复杂度，quantum 代表已知量子复杂度，plausible 代表可能存在的最好算法复杂度，m 代表所需样本数，b 代表 BKZ 算法分块大小。

Table 4: 高效参数攻击强度

参数(n, q, σ)	攻击类型	(m,b)	classical	quantum	plausible
f640:$(640, 2^{14}, 2.12)$	prime	(655,495)	145	131	103
	dual	(737,489)	143	130	101
f896:$(896, 2^{14}, 2.12)$	prime	(924,741)	216	196	153
	dual	(972,734)	214	195	153
f1216:$(1216, 2^{15}, 2.12)$	prime	(1199,976)	285	259	203
	dual	(1297,968)	283	257	201

6 性能分析

6.1 算法模块性能分析

公钥加密方案 SCloudCPAPKE 性能分析，算法包含密钥生成、加密、解密三个模块，算法运行开销主要可分为随机数生成器的调用、矩阵加法及乘法运算以及编码解码运算。

在密钥生成模块中，随机数生成器的调用需要生成 n^2 个模 q 均匀的随机数，进行 $2n\bar{n}$ 个采样；矩阵加法及乘法需要计算 1 次大小分别为 $n \times n$ 与 $n \times \bar{n}$ 的矩阵乘法以及 1 次大小为 $n \times \bar{n}$ 的矩阵加法。因此共需调用 $n^2 \log q + 2n\bar{n} \times E_\chi$ 比特的随机数，其中 E_χ 表示从分布χ 中采样需要的随机比特数，并进行 $n^2\bar{n}$ 次整数乘法以及 $n^2\bar{n}$ 次整数加法。

在加密模块中，随机数生成器的调用需要生成 n^2 个模 q 均匀的随机数，进行 $2\bar{m}n + \bar{m}\bar{n}$ 个采样；矩阵加法及乘法需要计算 1 次大小分别为 $\bar{m} \times n$ 与 $n \times n$ 的矩阵乘法, 1 次大小分别为 $\bar{m} \times n$ 与 $n \times \bar{n}$ 的矩阵乘法以及 1 次大小为 $\bar{m} \times n$

和 1 次大小为 $\bar{m} \times \bar{n}$ 的矩阵加法；编码解码运算需要进行 1 次编码操作。因此共需调用 $n^2 \log q + (2\bar{m}n + \bar{m}\bar{n}) \times E_\chi$ 比特的随机数，其中 E_χ 表示从分布 χ 中采样需要的随机比特数，并进行 $n^2\bar{m} + \bar{m}\bar{n}n$ 次整数乘法以及 $n^2\bar{m} + \bar{m}\bar{n}n$ 次整数加法，以及调用一次二元循环码的编码算法[2]，进行 $\bar{m}\bar{n}$ 次格雷码查表，再计算 $\bar{m}\bar{n}$ 次整数乘法。

在解密模块中，矩阵加法及乘法需要计算 1 次大小分别为 $\bar{m} \times n$ 与 $n \times \bar{n}$ 的矩阵乘法, 以及 1 次大小为 $\bar{m} \times \bar{n}$ 的矩阵加法；编码解码运算需要进行 1 次解码操作。因此共需进行 $\bar{m}\bar{n}n$ 次整数乘法，$\bar{m}\bar{n}n$ 次整数加法以及调用一次二元 BCH 码的译码算法[3]，进行 $\bar{m}\bar{n}$ 次格雷码查表，再计算 $\bar{m}\bar{n}$ 次整数除法。

随机数生成器产生随机比特串的速率与实现算法有关，本方案中使用 SM3 算法作为随机数生成器，根据输入长度的不同，软件实现的 SM3 算法输出随机比特的效率在 2-5 clock cycles/bit 之间[35]，以下按照 5 clock cycles/bit 进行计算。

综上，公钥加密方案 SCloudCPAPKE 高效参数效率见 Table 5 其输入输出长度见 Table 8。

密钥封装方案 SCloudKEM 性能分析，算法包含密钥生成、封装、解封三个模块，算法运行开销主要可分为随机数生成器的调用、矩阵加法及乘法运算以及编码解码运算。

在密钥生成模块中，随机数生成器的调用需要生成 n^2 个模 q 均匀的随机数，进行 $2n\bar{n}$ 个采样；矩阵加法及乘法需要计算 1 次大小分别为 $n \times n$ 与 $n \times \bar{n}$ 的矩阵乘法以及 1 次大小为 $n \times \bar{n}$ 的矩阵加法。因此共需调用 $n^2 \log q + 2n\bar{n} \times E_\chi$ 比特的随机数，其中 E_χ 表示从分布 χ 中采样需要的随机比特数，并进行 $n^2\bar{n}$ 次整数乘法以及 $n^2\bar{n}$ 次整数加法。

在封装模块中，随机数生成器的调用需要生成 n^2 个模 q 均匀的随机数，进行 $2\bar{m}n + \bar{m}\bar{n}$ 个采样；矩阵加法及乘法需要计算 1 次大小分别为 $\bar{m} \times n$ 与 $n \times n$ 的矩阵乘法, 1 次大小分别为 $\bar{m} \times n$ 与 $n \times \bar{n}$ 的矩阵乘法以及 1 次大小为 $\bar{m} \times n$ 和 1 次大小为 $\bar{m} \times \bar{n}$ 的矩阵加法；编码解码运算需要进行 1 次编码操作。因此

[2]约 $l_e(n_e - l_e)$ 次 Xor.

[3]约 $2 \cdot m_{ext} n_e(d_e - 1)$ 次 Xor [7]，这里 m_{ext} 为对应有限域的扩张次数。在本方案中，对于参数组 fg640, qg640, qg896 有 $m_{ext} = 8$，对于其它参数组 $m_{ext} = 9$。

共需调用 $n^2 \log q + (2\bar{m}n + \bar{m}\bar{n}) \times E_\chi$ 比特的随机数，其中 E_χ 表示从分布 χ 中采样需要的随机比特数，并进行 $n^2\bar{m} + \bar{m}\bar{n}n$ 次整数乘法以及 $n^2\bar{m} + \bar{m}\bar{n}n$ 次整数加法，以及调用一次二元循环码的编码算法，进行 $\bar{m}\bar{n}$ 次格雷码查表，再计算 $\bar{m}\bar{n}$ 次整数乘法。

在解封模块中，随机数生成器的调用需要生成 n^2 个模 q 均匀的随机数，进行 $2\bar{m}n + \bar{m}\bar{n}$ 个采样；矩阵加法及乘法需要计算 1 次大小分别为 $\bar{m} \times n$ 与 $n \times n$ 的矩阵乘法, 2 次大小分别为 $\bar{m} \times n$ 与 $n \times \bar{n}$ 的矩阵乘法以及 1 次大小为 $\bar{m} \times n$ 和 2 次大小为 $\bar{m} \times \bar{n}$ 的矩阵加法；编码解码运算需要进行 1 次编码和 1 次解码操作。因此共需调用 $n^2 \log q + (2\bar{m}n + \bar{m}\bar{n}) \times E_\chi$ 比特的随机数，其中 E_χ 表示从分布 χ 中采样需要的随机比特数，并进行 $n^2\bar{m} + 2\bar{m}\bar{n}n$ 次整数乘法以及 $n^2\bar{m} + 2\bar{m}\bar{n}n$ 次整数加法，以及调用一次二元循环码的编码算法和一次二元 BCH 码的译码算法，进行 $2\bar{m}\bar{n}$ 次格雷码查表，$\bar{m}\bar{n}$ 次整数乘法，再计算 $\bar{m}\bar{n}$ 次整数除法。

随机数生成器产生随机比特串的速率与实现算法有关，本方案中使用 SM3 算法作为随机数生成器，根据输入长度的不同，软件实现的 SM3 算法输出随机比特的效率在 2-5 clock cycles/bit 之间，以下按照 5 clock cycles/bit 进行计算。

综上，密钥封装方案 SCloudKEM 高效参数效率见 Table 6, 其输入输出长度见 Table 9。

公钥加密方案 SCloudCCAPKE 为封装方案加上一个DEM机制，我们使用的是加密消息为256位的AES-GCM加密认证算法，所以效率方面加密效率为SCloudKEM 的加密效率加AES-GCM加密效率，解密效率为SCloudKEM的解密效率加AES-GCM解密效率。SCloudCCAPKE加密方案的效率见 Table 7 ，其输入输出长度见 Table 10。

Table 5: SCloudCPAPKE 高效参数效率

参数组	$f640$	$f896$	$f1216$
密钥生成效率(clock cycle)	3.88×10^7	8.07×10^7	1.58×10^8
加密效率(clock cycle)	3.71×10^7	7.75×10^7	1.52×10^8
解密效率(clock cycle)	2.46×10^4	3.99×10^4	5.65×10^5

Table 6: SCloudKEM 高效参数效率

参数组	$f640$	$f896$	$f1216$
密钥生成效率(clock cycle)	3.88×10^7	8.07×10^7	1.58×10^7
封装效率(clock cycle)	3.71×10^7	7.75×10^7	1.52×10^8
解封效率(clock cycle)	3.71×10^7	7.75×10^7	1.52×10^8

Table 7: SCloudCCAPKE 高效参数效率

参数组	$f640$	$f896$	$f1216$
密钥生成效率(clock cycle)	1.88×10^7	4.14×10^7	8.07×10^7
加密效率(clock cycle)	1.71×10^7	3.82×10^7	7.48×10^7
解密效率(clock cycle)	1.71×10^7	3.82×10^7	7.48×10^7

Table 8: SCloudCPAPKE 高效参数输入输出长度

参数组	$f640$	$f896$	$f1216$
公钥长度(bit)	9.88×10^4	1.76×10^5	2.76×10^5
私钥长度(bit)	9.86×10^4	1.76×10^5	2.74×10^5
密文长度(bit)	8.2×10^4	1.53×10^5	2.40×10^5
明文长度(bit)	256	384	512

Table 9: SCloudKEM 高效参数输入输出长度

参数组	$f640$	$f896$	$f1216$
公钥长度(bit)	9.88×10^4	1.76×10^5	2.74×10^5
私钥长度(bit)	1.98×10^5	3.52×10^5	5.48×10^5
交换消息长度(bit)	8.23×10^4	1.53×10^5	2.40×10^5
协商密钥长度(bit)	256	384	512

Table 10: SCloudCCAPKE 高效参数输入输出长度

参数组	$f640$	$f896$	$f1216$
公钥长度(bit)	9.88×10^4	1.76×10^5	2.74×10^5
私钥长度(bit)	1.98×10^5	3.52×10^5	5.48×10^5
交换消息长度(bit)	8.25×10^4	1.53×10^5	2.41×10^5
协商密钥长度(bit)	256	384	512

6.2 与 Frodo 算法的对比

在表 11 中，我们给出了 SCloudKEM 在两组不同参数下与 FrodoKEM-640 [29]的效率对比。在我们的建议参数下，明文长度为安全级别的两倍，而 FrodoKEM-640 的明文长度与安全级别一致。因此我们给出了一组明文长度与安全级别一致的参数 SCloudKEM-fg640，见附录 E，并给出了其与FrodoKEM-640 的效率对比。SCloudKEM-fg640 的安全级别与 FrodoKEM-640 相同，均为 128 $bits$，且解密失败概率不低于 2^{-128}。此外，由于我们采用了国产密码算法标准 SM3 用作随机数源，而 FrodoKEM 采用了 AES 或 SHAKE 用作随机数源，因此在密钥生成、封装、解封效率上，我们集中在其所需乘法/加法次数上的对比。从Table 11中可以看到，在相同参数设置（SCloudKEM-fg640,FrodoKEM-640）下，我们的方案的参数规模与效率均优于Frodo。

Table 11: SCloudKEM 与 FrodoKEM 的传输规模与效率对比

	SCloudKEM -f640	SCloudKEM-fg640	FrodoKEM-640
公钥长度 (bits)	9.88×10^4	7.19×10^4	7.69×10^4
私钥长度 (bits)	1.97×10^5	1.44×10^5	1.59×10^5
明文长度 (bits)	256	128	128
密文长度 (bits)	8.23×10^4	6.38×10^4	7.78×10^4
密钥生成所需整数乘法/加法次数	4.51×10^6	3.28×10^6	3.28×10^6
封装所需整数乘法/加法次数	3.75×10^6	2.90×10^6	3.32×10^6
解封所需整数乘法/加法次数	3.81×10^6	2.94×10^6	3.36×10^6

7 适配性

1.方案设计中的随机数发生器以及Hash函数模块均使用国家密码管理局发布的密码散列函数标准SM3，算法可直接调用SM3算法模块，与其适配性良好。

2.在设计文档中给出的三个算法SCloudCPAPKE，SCloudCCAPKE 以及 SCloudKEM 之间存在相互转换关系（见图4）：（1）以SCloudCPAPKE 为基础，使用Fujisaki-Okamoto 转换以及哈希函数，可以构造SCloudKEM；（2）以SCloudKEM 为基础，使用DEM，可以构造SCloudCCAPKE。因此，SCloudCCAPKE 以及SCloudKEM 均可调用SCloudCPAPKE 的程序模块。同

时，SCloudCCAPKE 可直接调用SCloudKEM 为子程序。相同程序模块的使用减少了代码的书写量，提高代码的复用率，增强代码的可读性。

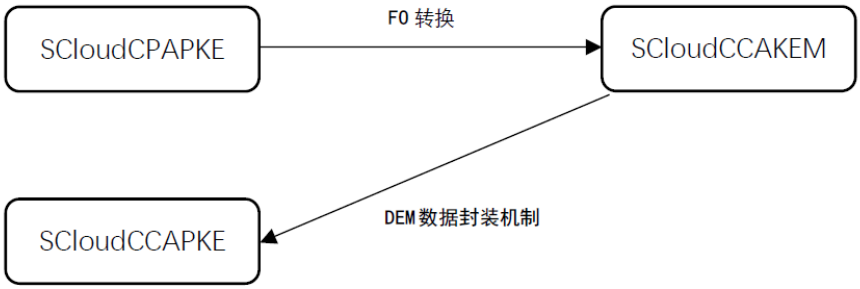

Figure 4: SCloud算法转换关系

8 优缺点

我们的方案有如下优点：

- 抗量子：我们的方案基于 LWE 问题，由于该问题基于的困难问题是无代数结构的随机格上的困难问题，而格上困难问题尚没有量子多项式算法，且较有代数结构困难问题（环上困难问题）有更强的安全度，所以我们的方案可以认为是抗量子的。
- 改进的采样算法：对比于其它通用的采样方式，我们的采样方法在保证安全性的基础上具有生成效率高的特点，且参数选取灵活。
- 应用纠错码技术来提高效率：我们的新型纠错方案-长纠错码结合格雷码，能够提供更高的传输及运算效率，同时使得我们的纠错方案的参数选取更加灵活。

同时，我们的方案有如下不足：

- 传输公私钥长度大：因为我们的方案基于 LWE 困难问题，相较有代数结构的方案，如基于 Ring-LWE 或 MLWE 问题方案，我们的参数设置会更

大。但对比同为基于 LWE 问题的 NIST 二轮方案 Frodo，在相同安全级别及相同加密消息比特下，我们的传输公私钥长度优于 Frodo，同时该参数规模可以应用到当下的应用环境中。

- 效率相对较低：因为我们的方案基于 LWE 困难问题，方案中的数据操作为矩阵上的运算，相较有代数结构的方案，如基于 Ring-LWE 或MLWE 问题方案，我们的方案效率要低一些。同样对比 Frodo，在相同安全级别及相同加密消息比特下，我们的效率优于 Frodo，同时效率在当下加解密环境仍然是高效的。

References

[1] Albrecht, M.R., Cid, C., Faugère, J., Fitzpatrick, R., Perret, L.: Algebraic algorithms for LWE problems. ACM Comm. Computer Algebra **49**(2), 62 (2015)

[2] Albrecht, M.R., Faugère, J., Fitzpatrick, R., Perret, L.: Lazy modulus switching for the BKW algorithm on LWE. In: Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings. pp. 429–445 (2014)

[3] Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Mathematical Cryptology **9**(3), 169–203 (2015)

[4] Andoni, A., Laarhoven, T., Razenshteyn, I.P., Waingarten, E.: Optimal hashing-based time-space trade-offs for approximate near neighbors. In: Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19. pp. 47–66 (2017)

[5] Arora, S., Ge, R.: New algorithms for learning in presence of errors. In: Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I. pp. 403–415 (2011)

[6] Baan, H., Bhattacharya, S., Fluhrer, S., Garcia-Morchon, O., Laarhoven, T., Player, R., Rietman, R., Saarinen, M.J.O., Tolhuizen, L., Arce, J.L.T., Zhang, Z.: Round5: compact and fast post-quantum public-key encryption,https://round5.org/

[7] Bajoga, B.G., Walbesser, W.J.: Decoder complexity for bch codes. Proceedings of the Institution of Electrical Engineers **120**(4), 429–431 (2010)

[8] Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016. pp. 10–24 (2016)

[9] Blum, A., Furst, M.L., Kearns, M.J., Lipton, R.J.: Cryptographic primitives based on hard learning problems. In: Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings. pp. 278–291 (1993)

[10] Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM **50**(4), 506–519 (2003)

[11] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012. pp. 309–325 (2012)

[12] Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings. pp. 1–20 (2011)

[13] Chunhuan, Z., Zhongxiang, Z., Xiaoyun, W., Guangwu, X.: Distinguishing lwe instances using fourier transform: A refined framework and its applications.https://eprint.iacr.org/

[14] Cramer, R., Ducas, L., Peikert, C., Regev, O.: Recovering short generators of principal ideals in cyclotomic rings. In: Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II. pp. 559–585 (2016). https://doi.org/10.1007/978-3-662-49896-5_20, `https://doi.org/10.1007/978-3-662-49896-5_20`

[15] Cramer, R., Ducas, L., Wesolowski, B.: Short stickelberger class relations and application to ideal-svp. In: Advances in Cryptology - EUROCRYPT 2017

- 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I. pp. 324–348 (2017)

[16] Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. **33**(1), 167–226 (2003)

[17] Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings. pp. 537–554 (1999)

[18] Guo, Q., Johansson, T., Mårtensson, E., Stankovski, P.: Coded-bkw with sieving. In: Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I. pp. 323–346 (2017)

[19] Guo, Q., Johansson, T., Stankovski, P.: Coded-bkw: Solving LWE using lattice codes. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I. pp. 23–42 (2015)

[20] Hanrot, G., Pujol, X., Stehlé, D.: Analyzing blockwise lattice algorithms using dynamical systems. In: Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings. pp. 447–464 (2011)

[21] Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the fujisaki-okamoto transformation. In: Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I. pp. 341–371 (2017)

[22] Kirchner, P., Fouque, P.: An improved BKW algorithm for LWE with applications to cryptography and lattices. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I. pp. 43–62 (2015). https://doi.org/10.1007/978-3-662-47989-6_3, https://doi.org/10.1007/978-3-662-47989-6_3

[23] Kirchner, P., Fouque, P.: Revisiting lattice attacks on overstretched NTRU parameters. In: Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I. pp. 3–26 (2017). https://doi.org/10.1007/978-3-319-56620-7_1, https://doi.org/10.1007/978-3-319-56620-7_1

[24] Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. Des. Codes Cryptography **75**(3), 565–599 (2015)

[25] Lindner, R., Peikert, C.: Better key sizes (and attacks) for lwe-based encryption. In: Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings. pp. 319–339 (2011). https://doi.org/10.1007/978-3-642-19074-2_21, https://doi.org/10.1007/978-3-642-19074-2_21

[26] Lu, X., Liu, Y., Zhang, Z., Jia, D., Xue, H., He, J., Li, B.: LAC: practical ring-lwe based public-key encryption with byte-level modulus. IACR Cryptology ePrint Archive **2018**, 1009 (2018), https://eprint.iacr.org/2018/1009

[27] Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings. pp. 1–23 (2010)

[28] Mariano, A., Laarhoven, T., Bischof, C.H.: A parallel variant of ldsieve for the SVP on lattices. In: 25th Euromicro International Conference on Parallel,

Distributed and Network-based Processing, PDP 2017, St. Petersburg, Russia, March 6-8, 2017. pp. 23–30 (2017)

[29] Naehrig, M., Alkim, E., Bos, J., Ducas, L., Easterbrook, K., LaMacchia, B., Longa, P., Mironov, I., Nikolaenko, V., Peikert, C., Raghunathan, A., Stebila, D.: Frodokem learning with errors key encapsulation. http://frodokem.org

[30] Peikert, C., Regev, O., Stephens-Davidowitz, N.: Pseudorandomness of ring-lwe for any ring and modulus. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017. pp. 461–473 (2017)

[31] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005. pp. 84–93 (2005)

[32] Schnorr, C.: A hierarchy of polynomial time lattice basis reduction algorithms. Theor. Comput. Sci. **53**, 201–224 (1987). https://doi.org/10.1016/0304-3975(87)90064-8, https://doi.org/10.1016/0304-3975(87)90064-8

[33] Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings. pp. 27–47 (2011). https://doi.org/10.1007/978-3-642-20465-4_4, https://doi.org/10.1007/978-3-642-20465-4_4

[34] Targhi, E.E., Unruh, D.: Post-quantum security of the fujisaki-okamoto and OAEP transforms. In: Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II. pp. 192–216 (2016)

[35] Wang, X. Yu, H.: Sm3 cryptographic hash algorithm. 2016

A 错误分布的区分优势分析

对随机变量$\mathbf{X}=(\mathbf{x}_1,\cdots,\mathbf{x}_n)$和向量$\mathbf{v}$，其中每个$\mathbf{x}_i$ 是在$\mathbb{Z}_q$上独立随机选取且均服从分布$f(x)$，我们考虑变量$\langle\mathbf{v},\mathbf{X}\rangle$的分布$f_{\langle\mathbf{v},\mathbf{X}\rangle}$. 根据傅立叶变换的性质，我们有

$$\hat{f}_{\langle\mathbf{v},\mathbf{X}\rangle}(1)=\prod_{j=1}^{n}\hat{f}_{v_j\mathbf{x}_j}(1)=\prod_{j=1}^{n}\hat{f}(v_j) \tag{2}$$

如果随机变量$\mathbf{X}\sim U(\mathbb{Z}_q^n)$，则由

$$\sum_{j=0}^{q-1}e^{\frac{-2\pi ijk}{q}}=0$$

可知$\hat{f}_{\langle\mathbf{v},\mathbf{X}\rangle}(1)=0$. 但是如果$\mathbf{x}_i$ 服从某种错误分布χ，对于合适选取的向量$\mathbf{v}$（即对$\mathbf{v}$的所有分量有$\hat{f}(v_j)\neq 0$），则会有$\hat{f}_{\langle\mathbf{v},\mathbf{X}\rangle}(1)\neq 0$，因此$|\hat{f}_{\langle\mathbf{v},\mathbf{X}\rangle}(1)|$可以作为区分$\langle\mathbf{v},\mathbf{X}\rangle$ 与$\mathbb{Z}_q$上均匀分布的优势，而此时对对偶格中的向量$\mathbf{v}$，根据(2),它所对应的区分优势与分量有关.

根据傅里叶变换的计算文献中有如下结果。

定理4.

- 对近似取整高斯分布$f(x)$，对$k\in\mathbb{Z}_q$有

$$\left|\widehat{f}(k)-\widehat{\Psi_{s,q}}(k)\right|\leq 2e^{\frac{-\pi R^2}{s^2}}.$$

其中

$$\frac{2}{\pi}e^{-\frac{\pi s^2k^2}{q^2}}\leqslant\widehat{\Psi_{s,q}}(k)\leqslant 2e^{-\frac{\pi s^2k^2}{q^2}}.$$

- 对服从中心二项分布$B(h)$的随机变量X，对$k\in\mathbb{Z}_q$，其对应的傅立叶变换为

$$\hat{f}(k)=\cos^{2h}(\frac{\pi k}{q}).$$

- 对随机变量$X\sim U[-a,-a+1,\cdots,0,1,\cdots,b]$，对$k=1,2,\cdots,\lfloor\frac{q}{2}\rfloor$，有

$$|\hat{f}(k)|^2=\frac{1-\cos\frac{2\pi(a+b+1)k}{q}}{(a+b+1)^2(1-\cos\frac{2\pi k}{q})}.$$

通过上述对分布的傅立叶变换分析，我们看到向量$\mathbf{v}$的分量会影响区分的复杂度. 现阶段无论错误分布具体的选取如何，衡量向量的区分优势均以$\epsilon(\|\mathbf{v}\|) = e^{-\frac{\pi s^2\|\mathbf{v}\|^2}{q^2}}$来估计，而以$\prod_{j=1}^{n}\hat{f}(v_j)$来估计体现出了不同分布的差异性，因此很自然的问题便是比较$\prod_{j=1}^{n}\hat{f}(v_j)$和$\epsilon(\|\mathbf{v}\|) = e^{-\frac{\pi s^2\|\mathbf{v}\|^2}{q^2}}$的大小，为此我们定义局部宽度$s(k)$.

定义13 (局部宽度). 对于$\mathbb{Z}_q$上的随机变量$X \sim f(x)$，当$1 \le k \le \lfloor \frac{q}{2} \rfloor$ 且$\hat{f}(k) \neq 0$，局部宽度$s(k)$ 定义为

$$s(k) = \frac{q}{k}\sqrt{\frac{-\ln|\hat{f}(k)|}{\pi}}.$$

局部宽度的定义是基于傅立叶变换估计，从向量的分量出发，反映具体分布的区分优势. 下面对几种分布的局部宽度和给定宽度的差距进行讨论.

定理5. *1.* 对中心二项分布$B(h)$，有

$$s(k)^2 \ge s^2 + 2\pi h\left(\frac{k\pi}{q}\right)^2\left(\frac{1}{12} + \frac{1}{45}\left(\frac{k\pi}{q}\right)^2 + \frac{17}{2520}\left(\frac{k\pi}{q}\right)^4\right),$$

其中$k = 1, 2, \cdots, \lfloor \frac{q}{2} \rfloor - 1$，$s = \sqrt{h\pi}$.

2. 对有界均匀分布$U[-a, -a+1, \cdots, 0, \cdots, b]$，其中$a + b \ge 7$，有

$$s(k)^2 + \frac{5(16qk - 3q^2)}{8k^2} < s^2,$$

其中$k \in [1, 2, \cdots, \lfloor q/2 \rfloor]$ 且满足条件$k \ge \frac{3q}{16\pi}$以及对任意整数ℓ, $|\frac{(a+b+1)k\pi}{q} - \ell\pi| \ge \frac{\pi}{24}$. 此处$s = \sqrt{\pi\frac{(a+b+1)^2-1}{6}}$.

因此对中心二项分布，局部宽度的下界为给定宽度，因此对对偶格中的向量使用$\epsilon = e^{-\pi s^2 l^2/q^2}$作为区分的优势，是保守的估计。而对于有界均匀分布，局部宽度在某些区间会低于给定宽度，进而说明了某些对偶格中向量所对应的区分能力是大于ϵ的。注意到对于近似取整高斯分布，上述定理并没有给出局部宽度和给定宽度理论上的关系，这是因为该比较会很大程度上依赖于采样的方

式、截断等因素，更直接地，可以对具体的近似分布进行具体计算进行分析。以采用上述三种分布的方案参数为例，给出局部宽度的特点。

- 提交给NIST 的Frodo 方案中错误服从的是近似取整高斯分布，以其中一组参数为例进行分析，即

$$(n, q, \sigma) = (640, 32768, 2.8),$$

错误服从的分布函数如表12 所示.

Table 12: Frodo-640中的错误分布函数

标准差	0	±1	±2	±3	±4	±5	±6	±7	±8	±9	±10	±11	±12
2.8	$\frac{9288}{65536}$	$\frac{8720}{65536}$	$\frac{7216}{65536}$	$\frac{5264}{65536}$	$\frac{3384}{65536}$	$\frac{1918}{65536}$	$\frac{958}{65536}$	$\frac{422}{65536}$	$\frac{164}{65536}$	$\frac{56}{65536}$	$\frac{17}{65536}$	$\frac{4}{65536}$	$\frac{1}{65536}$

可知分布的宽度$s_0 = \sigma\sqrt{2\pi} = 7.02$. 上述分布的局部宽度$s(k)$ 和s_0的比较在图5 中显示.

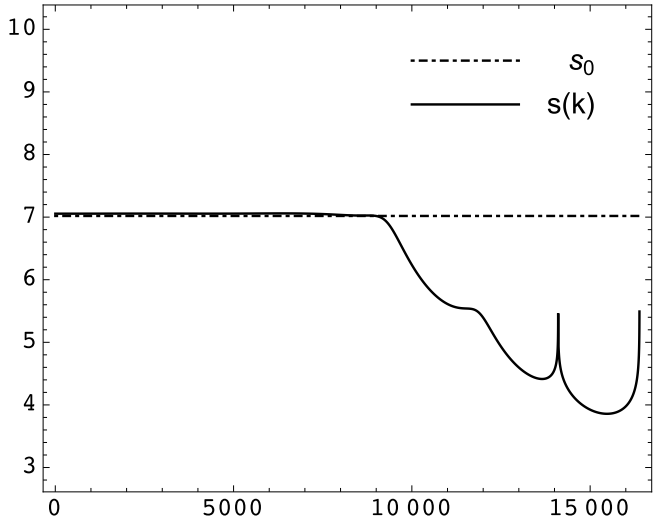

Figure 5: Frodo-640 中的局部宽度$s(k)$ 和给定宽度s_0的对比. 横坐标表示局部宽度中k的取值，纵坐标表示局部宽度和初始宽度的取值.

在图5 中可以看到局部宽度在长度大于$q/3$ 附近开始低于给定宽度，说明了对偶格中具有相同长度$\ell > q/3$ 的向量所对应的区分优势之间存在明显

的差距. 例如对长度为$q/2$的向量$\mathbf{v}$，如果存在分量长度为$q/3$, 则其区分优势的下界为

$$e^{-\pi(\frac{4}{9}s(\frac{q}{3})^2+\frac{5}{9}s_0^2)\|\mathbf{v}\|^2/q^2} \approx e^{-\pi(\frac{4*5.63^2}{9}+\frac{5*7.02^2}{9})\|v\|^2/q^2} \approx e^{-\pi 6.43^2\|v\|^2/q^2},$$

此时会明显大于$e^{-\pi s_0^2\|v\|^2/q^2}$. 另一方面，对于对偶格向量$\mathbf{v}$，如果它的每个分量长度都是小于$q/3$，则其对应的区分优势即为$e^{-\pi s_0^2\|v\|^2/q^2}$.

因此通过局部宽度的观察知，对偶格中合适的向量会对应较大的区分优势，从而较少的向量个数便可以用来区分. 另外，通过局部宽度的图像也得出了对应较大区分优势的对偶格向量的特点，即存在一个较大的分量.

- NewHope 方案是基于RLWE 问题构造的，其错误服从中心二项分布，同样地，我们以其中一组推荐参数为例进行分析，参数为

$$(n, q, \sigma) = (512, 12289, 2),$$

其中错误$e \sim B(8)$. 上述参数的局部宽度$s(k)$和给定宽度$s_0 = 2\sqrt{2\pi} \approx 5.01$之间的对比在图6 中显示.

可以看到，对所有的$k \leqslant q/2$，局部宽度$s(k)$始终大于给定宽度s_0，并且随着k的增大，差距$s(k) - s_0$也随之增大，这一点与定理5的结论一致. 另外，这说明了使用$e^{-\pi s_0^2\|\mathbf{v}\|^2/q^2}$作为区分优势进行估计是保守的.

- Saber 方案基于的是MLWR 问题. 该问题在评估时通常采用与LWE 问题相同的对偶区分模型进行计算，即以选取有界均匀采样作为错误分布. 我们以以下推荐的参数进行分析，即

$$(n, q, \sigma) = (768, 8192, 2.29),$$

其中错误$e \sim U[-3, -2, -1, 0, 1, 2, 3, 4]$. 上述参数的局部宽度$s(k)$ 和给定宽度$s_0 = 2\sqrt{2\pi} \approx 5.01$之间的对比在图7 中显示. 根据图7 中显示，当$k$大于某个数值时，$s(k)$ 和s_0 之间存在较大的差距. 特别地，当$k \geq q/5$时$s(k)$ 明显小于s_0，这与有界均匀分布的性质有关，与定理5 中的结果一致. 类似

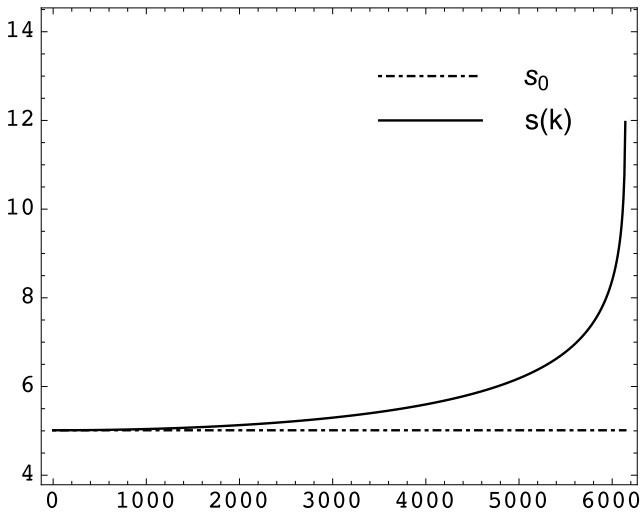

Figure 6: NewHope 方案中的局部宽度$s(k)$ 和给定宽度s_0的对比. 横坐标表示局部宽度中k的取值，纵坐标表示局部宽度和初始宽度的取值.

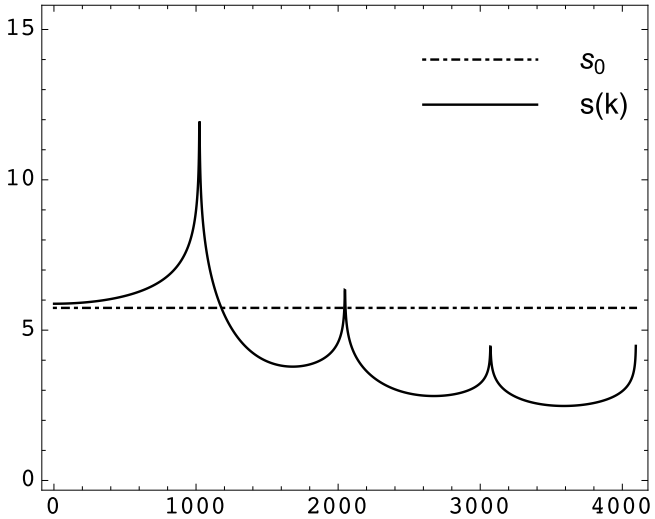

Figure 7: Saber 方案中的局部宽度$s(k)$ 和给定宽度s_0的对比. 横坐标表示局部宽度中k的取值，纵坐标表示局部宽度和初始宽度的取值.

地，$s(k)$小于s的部分说明了某些向量所对应的区分优势是明显大于现存的估计.

B 混合采样算法

根据混合采样的定义，可以证明采样的局部宽度始终大于给定宽度。

定理6. 令q为正整数，对于服从分布$\Phi(k_1, k_2)$的随机变量X，当$k_1 \geqslant k_2$时，对$j = 1, 2, \cdots, \lfloor q/2 \rfloor$, 有

$$s(j) \geqslant s_0,$$

其中$s_0 = \sqrt{(k_1 + \frac{4k_2}{3})\pi}$.

混合采样可以快速实现，可由中心二项分布和$\{-1, 0, 1\}$ 上的均匀分布作为底层算法，进行模块间的卷积，具体实现算法如下.

算法 1 – 混合采样算法

输入: 参数(k_1, k_2).

输出: 服从分布$\Phi(k_1, k_2)$的采样.

1: 输入参数(k_1, k_2)，表示采样算法的结果需要运行k_1 次二项分布采样模块与k_2 次整数均匀采样模块；

2: 输入2 比特随机源产生的随机数，输出服从中心二项分布的整数a，例如当随机数输入分别为00/01/10/11时，输出分别为$-1/0/0/1$；

3: 重复中心二项分布采样k_1 次，计算k_1 次输出之和，记第$i(i = 1, ,, k_1)$ 次输出为a_i，计算$A = a_1 + a_2 + \cdots + a_{k_1}$；

4: 输入$f = \lceil k_2 \log_2 3 \rceil$比特随机源产生的随机数，如果以二进制表示的随机数的数值大于3^{k_2}，则再次输入f 比特随机源产生的随机数，直到产生以二进制表示的随机数的数值不大于3^{k_2}的情况. 将输出的随机数表示为k_2 个三进制的字串，统计其中0 与2 的个数,分别为b_0，b_2，输出$B = b_2 - b_0$；

5: 输出采样结果$S = A + B \bmod q$.

算法的效率如下所示。

定理7. 算法1可以正确输出服从分布$\Phi_(k_1, k_2)$ 的采样，并且输出一个采样期望使用的随机源比特数为

$$2k_1 + \frac{\lceil k_2 \log_2 3 \rceil 2^{\lceil k_2 \log_2 3 \rceil}}{3^{k_2}}.$$

Table 14: 不同分布之间的局部宽度与效率对比

采样算法	效率	方差	局部宽度下界
混合采样	$2k_1 + f'2^{f'}/3^{k_2}$	$k_1/2 + 2k_2/3$	$\sqrt{(k_1 + 4k_2/3)\pi}$
近似离散高斯采样	$(\pi s^2 \log_2 e)/4$	$s^2/(2\pi)$	s
有界均匀分布	$f2^f/3^k$	$2k/3$	无较接近的下界
中心二项分布	$2k$	$k/2$	$\sqrt{k\pi}$

其中$f = \lceil k \log_2 3 \rceil$，$f' = \lceil k_2 \log_2 3 \rceil$.

在Table 14中给出各分布之间的方差，局部宽度与效率对比. 有界均匀分布采样的问题在于没有办法保证局部宽度存在较接近的下界，存在安全性隐患. 下面将主要就混合采样与可以有明确局部宽度的采样－近似离散高斯分布和中心二项分布进行对比，此时对比的重点主要是采样算法的效率以及采样范围. 图8 中给出了三种采样的效率对比，其中横坐标为局部宽度的数值，纵坐标为对应的采样效率.

C 代数攻击

代数攻击最早由 Arora 和 Ge 在[5] 中提出，它的基本思路是：构建以错误 e 为根的无噪音非线性多项式方程组，通过代换所有单项式构成线性方程组并求解。

具体来说，根据错误分布（高斯分布、小均匀分布等）的性质，我们可以假设错误 e 是落在区间 $[-t, t]$ （$t \in \mathbb{Z},\ d = 2t + 1 < q$）之间的。构造多项式 $P(x) = x \prod_{i=1}^{t}(x + i)(x - i)$，$e$ 是多项式 $P(x)$ 的根，则 s 是 $P(a \cdot x - b)$ 的根。在[5] 中，高次多项式方程组是通过替换单项式获得线性方程组求解；在改

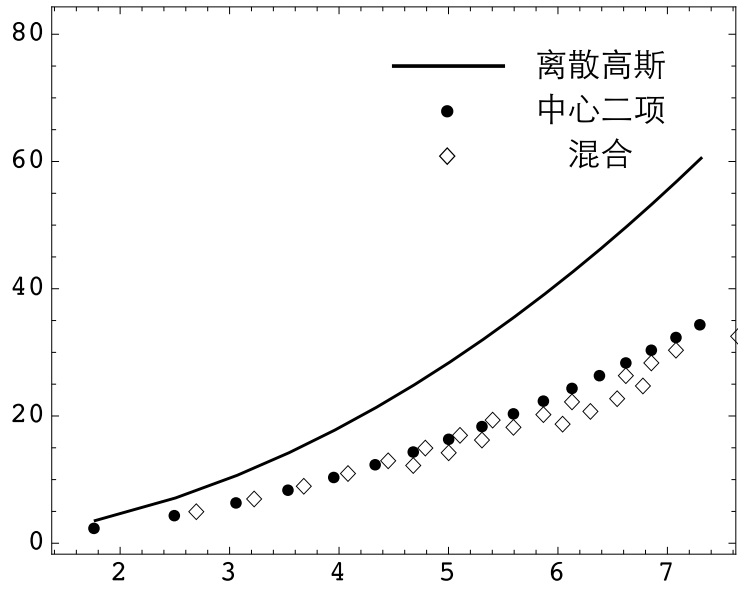

Figure 8: 采样算法的效率对比

进算法[1] 中，则使用 *Gröbner* 基来求解方程组。此时需要的方程数量减少，但是求解方程的时间复杂度增加。

定理8. （[1] 中的定理5）记 (n, q, σ) 为 LWE 实例的一组参数，$2 \leq \omega < 3$ 代表线性代数常数以及 $D_{AG} = 8\sigma^2 \log n + 1$。如果 $D_{AG} \in o(n)$，*Arora-Ge* 算法求解搜索版本的 LWE 时间复杂度为

$$O(2^{\omega \cdot D_{AG} \log \frac{n}{D_{AG}}} \cdot \sigma q \log q) = O(2^{8\omega\sigma^2 \log n(\log n - \log(8\sigma^2 \log n))} \cdot ploy(n)),$$

空间复杂度为

$$O(2^{2 \cdot D_{AG} \log \frac{n}{D_{AG}}} \cdot \sigma q \log q) = O(2^{16\sigma^2 \log n(\log n - \log(8\sigma^2 \log n))} \cdot ploy(n));$$

如果 $n \in o(D_{AG})$，*Arora-Ge* 算法求解搜索版本的 LWE 时间复杂度为

$$O(2^{\omega \cdot n \log \frac{D_{AG}}{n}} \cdot \sigma q \log q) = O(2^{\omega n \log(8\sigma^2 \log n) - \omega n \log n} \cdot ploy(n)),$$

空间复杂度为

$$O(2^{2\cdot n\log\frac{D_{AG}}{n}}\cdot\sigma q\log q)=O(2^{2n\log(8\sigma^2\log n)-2n\log n}\cdot ploy(n)).$$

我们以 $\sigma=\sqrt{n}$ 为例，给出两种算法的渐近复杂度估计。基本的 Arora-Ge 算法的渐近复杂度为 $O(2^{(2+\epsilon)\omega n\log\log n})$[3]。若结合 *Gröbner* 基，时间复杂度是 $O(2^{2.35\omega n+1.13n})$，存储复杂度是 $O(2^{5.85n})$ 以及需要的样本数量为 $m=\exp(\pi n/4)$[3]。

D BKW 攻击

BKW 算法最早由 Blum、Kalai 和 Wasserman 于[10] 中提出，它的基本思路是：借助广义生日攻击的方法在指数多的样本中挑选出少数几个样本使其加和为 0，此时零向量关于样本的系数向量即为对偶格中的短向量，可以用作 LWE 实例的区分器。

具体来说，BKW 构造短向量 $\mathbf{v}_i$ 的方法如下。给定一个样本 $\mathbf{A}\in M_{m\times n}$，BKW 分解行向量的 n 个分量为 a 个块，每个块的宽度为 $b=n/a$。算法共分成 a 个阶段，第 i 个阶段通过充分多的上一级样本碰撞（加法运算）获得第 i 块元素全零的样本。假设初始样本数量为 2^aq^b，在最优的情况下，算法结束时可获得 q^b 个长度为 $\sqrt{2^a}$ 的短向量。

接下来，使用短向量做区分攻击的方法同对偶攻击中的区分攻击是相同的。每个短向量的区分概率为 $\varepsilon=\exp(-2\pi^2\cdot 2^a(\frac{\sigma}{q})^2)$，大约需要 $1/\varepsilon^2$ 个短向量来放大成功率。因为一次 BKZ 算法可生成 q^b 个向量，所以选取参数 a 时需满足 $1/(\exp(-2\pi^2\cdot 2^a(\frac{\sigma}{q})^2))^2\leq q^{n/a}$ 条件。

从 BKW 提出以来，有两个主要的改进算法：一个是 Coded-BKW 算法，最初的想法由 Albrecht 等人在 PKC 2014[2] 上提出，他们使用 Lazy Modulus Switching 技术求解小密钥版本的 LWE 问题。次年，[19] 和[22] 同时独立的提出了思想相近的算法用于改进 BKW，这里沿用[19] 中的名词称其为 Coded-BKW 算法；另一个是在 Coded-BKW 算法的基础上结合筛法进行改进[18]，目前最优的时间复杂度是结合筛法得到的。

定理9. [18] 记 (n, q, σ) 为 LWE 实例的一组参数，有 $q = n^{c_q}, \sigma = n^{c_s}$。Coded-BKW 结合筛选算法的时间复杂度以及空间复杂度均为 $2^{(c+o(1))n}$，其中

$$c = \frac{1}{1 - e^{-\lambda(1+2(c_q - c_s))/c_s}},$$

在经典计算机下 $\lambda = 0.292$ 以及量子计算机下 $\lambda = 0.265$。

定理10. [18] 如果$c > \lambda$ 且$\frac{c_s}{\lambda} \ln \frac{c_q}{c_s} < 2(c_q - c_s) + 1$，那么，使用平凡的 BKW 做预处理后 Coded-BKW 结合筛选算法的时间复杂度以及空间复杂度均为 $2^{(c+o(1))n}$，其中

$$c = (\lambda^{-1}(1 - \frac{c_s}{c_q}) + c_q^{-1}(2(c_q - c_s) + 1 - \frac{c_s}{\lambda} \ln \frac{c_q}{c_s}))^{-1}$$

在经典计算机下 $\lambda = 0.292$ 以及量子计算机下 $\lambda = 0.265$。

使用 Regev 的 LWE 实例参数时，$q = n^2$ 且 $\sigma = n^{1.5}/(\sqrt{2\pi} \log_2^2 n)$，带预处理且结合筛法的 BKW 算法渐近复杂度为 $2^{0.895n}$，是目前最好的理论结果。

在我们设计的方案中：（1）在高效参数方案中会定时更换密钥，敌手无法获得代数攻击所需要的样本数量，因此 BKW 攻击无法实施。（2）在量子安全算法下，我们根据定理6 给出不同参数相应结合筛法的 BKW 算法攻击复杂度，如 Table 6 所示：

E 其他参数

E.1 量子安全参数

我们给出一组量子安全的参数，见Table 15，该参数下高斯标准差至少为 $\frac{\sqrt{n}}{2\pi}$[30]，从而有安全规约。并给出相应的安全强度，原始攻击与对偶攻击见Table 16，BKW类型攻击见Table 17,代数攻击见Table 18。

在快速算法下，方案会定时更换密钥，敌手无法获得代数攻击所需要的样本数量，因此代数攻击无法实施；在量子安全算法下，我们根据定理4 给出不同参数相应 Arora-Ge 代数攻击复杂度，如下表所示：

Table 15: 量子安全的保守参数选取

	q640	q896	q1216
安全等级（bit）	128	192	256
n	640	896	1216
q	2^{16}	2^{17}	2^{18}
$(\bar{m}, \bar{n})$	$(9, 11)$	$(12, 12)$	$(13, 14)$
σ	4.10	4.83	5.85
消息m比特数	256	384	512
(k_1, k_2) in χ_{k_1,k_2}	(15,14)	(20,20)	(30,29)
解密失败概率	2^{-135}	2^{-206}	2^{-295}
$[n_e, l_e, d_e]$ 线性码	[292,256,9]	[474,384,21]	[552,512,9]
h-比特格雷码	3	3	3

E.2 短消息参数

我们在这里提供两组在SCloudCPAPKE中加密的密钥比特长度为对应安全强度比特位的参数，如128比特安全强度参数其加密消息也为128比特。我们给出一组量子安全的参数，该参数下高斯标准差至少为 $\frac{\sqrt{n}}{2\pi}$[30]，从而有安全规约，参数见Table 19，另一组参数为高效参数，其沿袭我们文中给出的思路，虽然高斯标准差没有达到规约安全度的大小，但是在现有攻击模型下，此参数仍满足抵抗原始攻击和对偶攻击的安全强度要求，高效参数见Tabel 20。对应攻击复杂度见Table 21,22,23,24。

Table 16: 保守参数攻击强度

参数(n, q, σ)	攻击类型	(m,b)	classical	quantum	plausible
q640:$(640, 2^{16}, 4.10)$	prime	(737,484)	141	128	100
	dual	(832,479)	140	127	99
q896:$(896, 2^{17}, 4.83)$	prime	(1026,691)	202	183	143
	dual	(1143,685)	200	182	142
q1216:$(1216, 2^{18}, 5.85)$	prime	(1396,959)	280	254	199
	dual	(1494,951)	278	252	197

Table 17: 结合筛法的BKW 攻击的复杂度估计

	q640	q896	q1216
安全等级（bit）	128	192	256
参数(n, q, σ)	$(640, 2^{16}, 4.10)$	$(896, 2^{17}, 4.83)$	$(1216, 2^{18}, 5.85)$
classical	2^{145}	2^{206}	2^{284}
quantum	2^{138}	2^{196}	2^{271}

Table 18: 代数攻击的复杂度估计

	q640	q896	q1216
安全等级（bit）	128	192	256
参数(n, q, σ)	$(640, 2^{16}, 4.1)$	$(896, 2^{17}, 4.83)$	$(1216, 2^{18}, 5.85)$
时间复杂度	$O(2^{620.8\omega})$	$O(2^{923.4\omega})$	$O(2^{1467\omega})$
存储复杂度	$O(2^{1242})$	$O(2^{1847})$	$O(2^{2933})$

Table 19: 量子安全的保守参数选取

	qg640	qg896	qg1216
安全等级（bit）	128	192	256
n	640	896	1216
q	2^{16}	2^{17}	2^{18}
$(\bar{m}, \bar{n})$	$(6, 9)$	$(8, 9)$	$(8, 12)$
σ	4.10	4.83	5.85
消息m比特数	128	192	256
(k_1, k_2) in χ_{k_1,k_2}	(15,14)	(20,20)	(30,29)
解密失败概率	2^{-128}	2^{-192}	2^{-264}
$[n_e, l_e, d_e]$ 线性码	[152,128,7]	[216,192,7]	[283,256,7]
h-比特格雷码	3	3	3

Table 20: 高效的参数选取

	fg640	fg896	fg1216
安全等级（bit）	128	192	256
n	640	896	1216
q	2^{14}	2^{14}	2^{15}
$(\bar{m}, \bar{n})$	$(7, 8)$	$(8, 12)$	$(9, 11)$
σ	2.12	2.12	2.12
消息m比特数	128	192	256
(k_1, k_2) in χ_{k_1,k_2}	(5,3)	(5,3)	(5,3)
解密失败概率	2^{-168}	2^{-214}	2^{-299}
$[n_e, l_e, d_e]$ 线性码	[168,128,11]	[282,192,21]	[292,256,9]
h-比特格雷码	3	3	3

Table 21: 保守参数攻击强度

参数(n, q, σ)	攻击类型	(m,b)	classical	quantum	plausible
qg640:$(640, 2^{16}, 4.10)$	prime	(737,484)	141	128	100
	dual	(832,479)	140	127	99
qg896:$(896, 2^{17}, 4.83)$	prime	(1026,691)	202	183	143
	dual	(1143,685)	200	182	142
qg1216:$(1216, 2^{18}, 5.85)$	prime	(1396,959)	280	254	199
	dual	(1494,951)	278	252	197

Table 22: 高效参数攻击强度

参数(n, q, σ)	攻击类型	(m,b)	classical	quantum	plausible
fg640:$(640, 2^{14}, 2.12)$	prime	(655,495)	145	131	103
	dual	(737,489)	143	130	101
fg896:$(896, 2^{14}, 2.12)$	prime	(924,741)	216	196	153
	dual	(972,734)	214	195	153
fg1216:$(1216, 2^{15}, 2.12)$	prime	(1199,976)	285	259	203
	dual	(1297,968)	283	257	201

Table 23: 结合筛法的BKW 攻击的复杂度估计

	qg640	qg896	qg1216
安全等级（bit）	128	192	256
参数(n, q, σ)	$(640, 2^{16}, 4.10)$	$(896, 2^{17}, 4.83)$	$(1216, 2^{18}, 5.85)$
classical	2^{145}	2^{206}	2^{284}
quantum	2^{138}	2^{196}	2^{271}

Table 24: 代数攻击的复杂度估计

	qg640	qg896	qg1216
安全等级（bit）	128	192	256
参数(n, q, σ)	$(640, 2^{16}, 4.1)$	$(896, 2^{17}, 4.83)$	$(1216, 2^{18}, 5.85)$
时间复杂度	$O(2^{620.8\omega})$	$O(2^{923.4\omega})$	$O(2^{1467\omega})$
存储复杂度	$O(2^{1242})$	$O(2^{1847})$	$O(2^{2933})$