LAC.KEX

LAttice-based **C**ryptosystems: Key Exchange

基于格的密码系统:密钥交换

第一设计者: 路献辉[1,2], 15010131069, luxianhui@iie.ac.cn

其他参与者: 刘亚敏[1,2], liuyamin@iie.ac.cn

贾仃仃[1,2], jiadingding@iie.ac.cn

薛海洋[1,2], xuehaiyang@iie.ac.cn

贺婧楠[1,2], hejingnan@iie.ac.cn

张振飞[3], zhenfei@algorand.com

李宝[1,2], libao@iie.ac.cn

王鲲鹏[1,2], wangkunpeng@iie.ac.cn

参与者单位: 1. 中国科学院数据与通信保护研究教育中心

2. 中国科学院信息工程研究所信息安全国家重点实验室

3. Algorand

算法联系人: 路献辉

通信地址： 北京市海淀区闵庄路甲89号中国科学院信息工程研究所

1 概览

我们基于格上的环LWE(Learning with errors over rings)假设构造了LAC密码系统(**LA**ttice-based **C**ryptosystems), 包含下述四个公钥密码算法。根据中国密码学会密码算法竞赛规则，为了评估方便，LAC系统拆分为LAC.PKE和LAC.KEX两个部分进行分别评估。本文档介绍LAC.KEX部分，包含基于LAC.PKE设计的基础密钥交换算法LAC.KEX和基于选择密文安全的LAC.KEM设计的认证密钥交换算法LAC.AKE。

- LAC.PKE: IND-CPA安全的公钥加密方案;
- LAC.KEX: 被动安全的密钥交换协议, 从LAC.PKE直接转换而来;
- LAC.KEM: IND-CCA安全的密钥封装机制, 是对LAC.PKE应用文献[17,19,20]中的FO转换变形而来;
- LAC.AKE: 认证密钥交换协议, 是对LAC.KEM和LAC.PKE应用[15,16]中的FSXY转换而来.

图1中示出了这四个算法之间的关系.

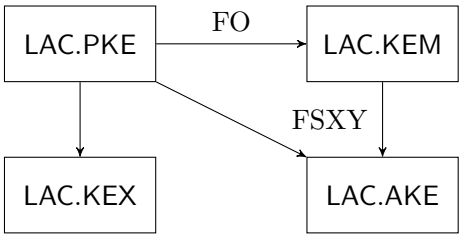

图 1. 基于格的密码系统LAC

原创性和版本说明. LAC 密码系统是LAC 提交者和各位参与者的原创工作, 其设计和实现没有申请专利, 也不受任何已知专利的覆盖. LAC 的代码实现中使用了公开免费的库和公开授权可在非商业用途中免费使用及修改的第三方代码.

LAC 的第一个版本[22]作为候选算法提交到美国NIST组织的全球后量子公钥密码算法标准化项目中, 并通过了初筛和第一轮评估, 进入了第二轮评估. 在此期间, 我们修改了算法的消息长度, 并公布在[23], 基于次改进进的版本提交到我国密码算法竞赛, 根据我国密码算法竞赛第一轮的评审结果, 我们对算法进行了修改之后形成提交我国密码算法竞赛的第二轮版本，主要修改如下：

- 使用新的BCH参数,并配合使用D2纠错算法将解密错误率降低到安全级别以下;
- 将公钥pk用于加密随机数的产生, 更好地抵抗多密钥选择密文攻击；
- 噪音采样、多项式乘法、BCH解码均给出实现常数时间实现；
- 增加了LAC-light参数面向轻量级应用；
- 增加了LAC-test参数作为指定参数供测评使用。

文档组织. 本文档组织结构如下:

第2部分定义了数学记号和运算，并介绍了关于格的背景知识; 第3部分介绍了总体的设计原理; 第4部分包含了算法的详细描述; 第5部分给出了算法的参数设置, 其中包括对算法的解密错误率分析; 第6部分给出了安全性分析; 第7部分描述了算法的代码实现, 性能参数, 以及测试数据生成方式; 最后，第8部分分析了算法的优缺点.

2 预备知识

这一部分给出了文档中所用的记号和运算的说明, 以及关于格的背景知识.

2.1 数学记号

向量和矩阵. 向量用小写黑体字母表示, 例如$\boldsymbol{a}$. 向量$\boldsymbol{a}$的转置用$\boldsymbol{a}^t$表示.

定义m维向量$\boldsymbol{a} = (a_0, \cdots, a_{m-1})$, 其中对于$0 \le i < m$, a_i是$\boldsymbol{a}$的各个分量.

对于标量s和m维向量$\boldsymbol{a}$, 以$s \cdot \boldsymbol{a}$表示其乘积, 其中$\boldsymbol{a}$的每个分量都乘以s, 即, $s \cdot \boldsymbol{a} = (s \cdot a_0, \cdots, s \cdot a_{m-1})$.

对于m维向量$\boldsymbol{a} = (a_0, \cdots, a_{m-1})$和非负整数$l \le m$, 定义$(\boldsymbol{a})_l = (a_0, \cdots, a_{l-1})$.

相同维度的向量可以逐分量相加, 例如, 对于两个m维向量$\boldsymbol{a} = (a_0, \cdots, a_{m-1})$和$\boldsymbol{b} = (b_0, \cdots, b_{m-1})$, $\boldsymbol{a} + \boldsymbol{b} = (a_0 + b_0, \cdots, a_{m-1} + b_{m-1})$.

矩阵用大写黑体字母表示, 例如$\boldsymbol{A}$. 矩阵$\boldsymbol{A}$的转置用$\boldsymbol{A}^t$表示.

范数. 向量长度用范数来度量. 对于m维向量$\boldsymbol{x} = (x_1, x_2, ..., x_m)$, 其$l_1$范数定义为$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{m} |x_i|$; l_2范数, 即欧几里得范数, 定义为$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{m} x_i^2}$, 或简单记为$\|\boldsymbol{x}\|$; 无穷范数定义为$\|\boldsymbol{x}\|_\infty = \max |x_i|$. 矩阵的长度定义为其最长的列向量的范数, 即, $\|\boldsymbol{X}\| = \max \|\boldsymbol{x}_i\|$.

箭头. 对于集合S, 以$x \xleftarrow{\$} S$表示从$S$中均匀随机地选择元素$x$. 对于分布$D$, 以$x \xleftarrow{\$} D$表示从$D$中采样随机变量$x$. 对于随机算法$\mathsf{A}$, 以$y \xleftarrow{\$} \mathsf{A}(x)$表示将$\mathsf{A}$在输入$x$上的输出赋值给$y$; 若算法$\mathsf{A}$是确定的, 则以$y \leftarrow \mathsf{A}(x)$表示.

代数结构. 令$\mathbb{R}$表示实数, $\mathbb{Q}$表示有理数, $\mathbb{Z}$表示整数. 对于整数$q \ge 1$, 令$\mathbb{Z}_q$为模q的剩余类环, 且$\mathbb{Z}_q = \{0, \cdots, q-1\}$. 对于整数$n \ge 1$, 定义模$x^n + 1$的整数多项式环为$R = \mathbb{Z}[x]/(x^n + 1)$, 定义系数来自$\mathbb{Z}_q$的多项式环$R_q = \mathbb{Z}_q[x]/(x^n + 1)$. R_q中元素的加法和乘法运算遵守多项式的运算规则.

字符串运算. 对于两个比特串$s_1, s_2 \in \{0,1\}^*$, 定义它们的拼接为$s_1 \| s_2$. 定义比特串s的长度为$|s|$. 有时我们将长度为m的比特串视为m维向量. 空串用ϵ来表示.

2.2 格及其困难问题

格与对偶格. 定义由基$\boldsymbol{B} = \{\boldsymbol{b}_1, ..., \boldsymbol{b}_m\} \in \mathbb{R}^{m \times m}$生成的$m$维满秩格$\Lambda$为

$$\Lambda = \mathcal{L}(\boldsymbol{B}) = \{\boldsymbol{B}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{Z}^m\},$$

其中$\boldsymbol{b}_1, ..., \boldsymbol{b}_m$为线性无关的向量.

对于m维满秩格Λ, 定义其对偶格为

$$\Lambda^* = \{\boldsymbol{y} \in \mathbb{R}^m : \forall \boldsymbol{x} \in \Lambda, \langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}\}.$$

q元格. 大多数格密码方案基于两类特殊的满秩格, 即q元整数格. 对于正整数n, m, q和随机整数矩阵$\boldsymbol{A} \in \mathbb{Z}_q^{m\times n}$, 定义如下$m$维满秩$q$元整数格:

$$\Lambda(\boldsymbol{A}) = \{\boldsymbol{z} \in \mathbb{Z}^m : \exists \boldsymbol{s} \in \mathbb{Z}_q^n \text{ s.t. } \boldsymbol{z} = \boldsymbol{A}\boldsymbol{s} \bmod q\};$$

$$\Lambda^{\perp}(\boldsymbol{A}^t) = \{\boldsymbol{z} \in \mathbb{Z}^m : \boldsymbol{A}^t\boldsymbol{z} = \boldsymbol{0} \bmod q\}.$$

格困难问题. 定义格Λ中最短非零向量的长度为$\lambda_1(\Lambda)$. 格上最基本的计算问题即最短向量问题(SVP). 通常使用的是SVP问题关于参数$\gamma \geq 1$的近似变形.

定义 1 (搜索SVP$_\gamma$) *给定格基$\boldsymbol{B} \in \mathbb{Z}^{m\times n}$和$\gamma \geq 1$, 找到$\boldsymbol{v} \in \mathcal{L}(\boldsymbol{B})$使$\|\boldsymbol{v}\| \leq \gamma \cdot \lambda_1(\boldsymbol{B})$.*

定义 2 (GapSVP$_\gamma$) *对于$\gamma \geq 1$, 给定输入对$(\boldsymbol{B}, r)$, 其中$\boldsymbol{B} \in \mathbb{Z}^{m\times m}$为$m$维满秩格的基, 且$r \in \mathbb{Q}$. 判断: 若$\lambda_1(\mathcal{L}(\boldsymbol{B})) \leq r$则称其为YES实例, 若$\lambda_1(\mathcal{L}(\boldsymbol{B})) > \gamma \cdot r$则称其为NO实例.*

作为对λ_1的推广, 定义第i连续最小量$\lambda_i(\Lambda)$为一个最小半径r, 使得Λ包含i个长度最多为r的线性无关向量. 相关的最短无关向量问题(SIVP)及其近似版本, 以及$\gamma-$uSVP问题定义如下.

定义 3 (SIVP) *给定m维满秩格的格基$\boldsymbol{B} \in \mathbb{Z}^{m\times m}$, 输出$m$个线性无关的格向量的集合$\boldsymbol{S} = \{\boldsymbol{s}_1, ..., \boldsymbol{s}_m\} \subset \mathcal{L}(\mathbb{B})$, 使得$\|\boldsymbol{s}_i\| = \lambda_i(\mathcal{L}(\boldsymbol{B}))$.*

定义 4 (SIVP$_\gamma$) *对于$\gamma \geq 1$, 给定m维满秩格的格基$\boldsymbol{B} \in \mathbb{Z}^{m\times m}$, 输出$m$个线性无关的格向量的集合$\boldsymbol{S} \subset \mathcal{L}(\mathbb{B})$, 使得$\|\boldsymbol{S}\| \leq \gamma \cdot \lambda_n(\mathcal{L}(\boldsymbol{B}))$.*

定义 5 ($\gamma-$uSVP) *对于$\gamma \geq 1$. 当格Λ满足$\lambda_2(\Lambda) > \gamma\lambda_1(\Lambda)$ 时, 寻找非零向量$\boldsymbol{u} \in \Lambda$, 使得$\|\boldsymbol{u}\| = \min_{\boldsymbol{v}} \|\boldsymbol{v}\|$.*

2.3 (环)带错误的学习问题

目前大部分基于格的公钥加密算法和密钥交换协议是基于带错误的学习问题(LWE)[28]及其变形构造的. 我们提出的LAC密码系统基于环LWE[24]的简单版本.

定义 6 (搜索LWE) *令n, m, q为正整数, $\chi_{\boldsymbol{s}}, \chi_{\boldsymbol{e}}$ 为$\mathbb{Z}$上的分布. 给定$(\boldsymbol{A}, \boldsymbol{b} = \boldsymbol{A}\boldsymbol{s}+\boldsymbol{e})$, 求秘密向量$\boldsymbol{s}$. 其中$\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, 秘密向量$\boldsymbol{s} \xleftarrow{\$} \chi_{\boldsymbol{s}}^n$, 错误向量$\boldsymbol{e} \xleftarrow{\$} \chi_{\boldsymbol{e}}^m$.*

LWE假设的合理性基于格上的困难问题, 即此前所述的GapSVP$_\gamma$和SIVP$_\gamma$问题, 其中γ的选择与参数n, m, q以及秘密和错误向量的分布$\chi_{\boldsymbol{s}}, \chi_{\boldsymbol{e}}$相关.

定义 7 (判定LWE) *令n, m, q为正整数, $\chi_{\boldsymbol{s}}, \chi_{\boldsymbol{e}}$为$\mathbb{Z}$上的分布. 区分以下两个分布:*

- $D_0 : (\boldsymbol{A}, \boldsymbol{b})$, *与*
- $D_1 : (\boldsymbol{A}, \boldsymbol{u})$,

其中$\boldsymbol{b} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}$, $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, $\boldsymbol{s} \xleftarrow{\$} \chi_{\boldsymbol{s}}^n$, $\boldsymbol{e} \xleftarrow{\$} \chi_{\boldsymbol{e}}^m$, $\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_q^m$.

LWE的判定版本和计算版本是多项式等价的[25].

在环LWE中, 带噪音的方程是$(\boldsymbol{a}, \boldsymbol{b} = \boldsymbol{a}\boldsymbol{s} + \boldsymbol{e})$, 其中$\boldsymbol{a}$, $\boldsymbol{s}$, $\boldsymbol{e}$ 为环元素. 常用的环是整数多项式环$R_q = \mathbb{Z}_q[x]/(x^n + 1)$, 其中$n$是合适的环维度. 有时$R_q$上的环LWE特例称为poly-LWE [9], 其中以$v \xleftarrow{\$} \chi$ 表示$v \in R$的每个系数均按照分布χ生成. 在LAC中我们使用poly-LWE. 环LWE假设的合理性基于理想格上的SVP_γ问题, 而不是随机格.

简单起见, 我们使用环LWE最常用的定义, 即多项式环$R_q = \mathbb{Z}_q[x]/(x^n + 1)$上的poly-LWE.

定义 8 (搜索环LWE) 令n, q为正整数, $\chi_{\boldsymbol{s}}, \chi_{\boldsymbol{e}}$为$R$上的分布. 给定$(\boldsymbol{a}, \boldsymbol{b} = \boldsymbol{a}\boldsymbol{s} + \boldsymbol{e})$, 求秘密$\boldsymbol{s}$, 其中$\boldsymbol{a} \xleftarrow{\$} R_q$, 秘密$\boldsymbol{s} \xleftarrow{\$} \chi_{\boldsymbol{s}}$, 错误$\boldsymbol{e} \xleftarrow{\$} \chi_{\boldsymbol{e}}$.

定义 9 (判定环LWE) 令n, q为正整数, $\chi_{\boldsymbol{s}}, \chi_{\boldsymbol{e}}$为$R$上的分布. 区分以下两个分布:

- $D_0 : (\boldsymbol{a}, \boldsymbol{b})$, 与
- $D_1 : (\boldsymbol{a}, \boldsymbol{u})$,

其中$\boldsymbol{b} = \boldsymbol{a}\boldsymbol{s} + \boldsymbol{e}$, $\boldsymbol{a} \xleftarrow{\$} R_q$, $\boldsymbol{s} \xleftarrow{\$} \chi_{\boldsymbol{s}}$, $\boldsymbol{e} \xleftarrow{\$} \chi_{\boldsymbol{e}}$, $\boldsymbol{u} \xleftarrow{\$} R_q$.

2.4 分布和随机采样

均匀分布. 定义集合X上的均匀分布为$U(X)$. 例如, R_q上的均匀分布为$U(R_q)$.

高斯分布. (环)LWE中的秘密和错误通常采样自高斯分布. 一般的高斯分布是连续分布, 其概率密度函数定义如下:

$$\rho_\sigma(x) = (\sqrt{2\pi}\sigma)^{-1} \exp(-\pi x^2/2\pi\sigma^2),$$

其中σ为标准差.

$\mathbb{Z}$上的离散高斯分布定义为:

$$\forall x \in \mathbb{Z}, \mathcal{D}_{\mathbb{Z},\sigma}(x) = \frac{\rho_\sigma(x)}{\rho_\sigma(\mathbb{Z})},$$

其中$\rho_\sigma(\mathbb{Z}) = \sum_{y\in\mathbb{Z}} \rho_\sigma(y)$.

$\mathbb{Z}_q$上的离散高斯分布定义为:

$$\forall x \in \mathbb{Z}_q, \mathcal{D}_{\mathbb{Z}_q,\sigma}(x) = \sum_{w, w=x \bmod q} \mathcal{D}_{\mathbb{Z},\sigma}(w).$$

中心二项分布. 由于高斯分布的采样并非易事, 文献[5]中提出在设计实用型格密码系统时可使用中心二项分布. 令Ψ_σ为中心二项分布, 其参数为σ, 对应的标准差为$\sqrt{\frac{\sigma}{2}}$. 在LAC的设计中, 我们使用参数为1和$\frac{1}{2}$的中心二项分布, 分别记为Ψ_1和$\Psi_{\frac{1}{2}}$. 其定义如下:

定义 10 (Ψ_1) *采样*$(a,b) \xleftarrow{\$} \{0,1\}^2$*, 并输出*$a-b$*. 显然输出*0*的概率为*$\frac{1}{2}$*, 输出*$\pm1$*的概率均为*$\frac{1}{4}$*.* Ψ_1*的均值为*0*, 方差为*$\frac{1}{2}$*.*

定义 11 ($\Psi_{\frac{1}{2}}$) *采样*$(a,b) \xleftarrow{\$} \Psi_1$*, 并输出*$a*b$*. 显然输出*0*的概率为*$\frac{3}{4}$*, 输出*$\pm1$*的概率均为*$\frac{1}{8}$*.* $\Psi_{\frac{1}{2}}$*的均值为*0*, 方差为*$\frac{1}{4}$*.*

对正整数n, 以Ψ_σ^n表示n重独立同分布的Ψ_σ, 其中σ是分布的参数. 在按照分布Ψ_σ^n采样时, 随机变量的n的分量的选取都是独立的.

此外, 我们还定义固定汉明重量的n重中心二项分布, 记为Ψ_n^h, 其中$0 < h < n/2$为偶数. 对于服从此分布的n重随机变量, 其汉明重量固定为其期望值h, 且其中值为1和-1的分量个数均为$h/2$个, 值为0的分量个数为$n-h$.

随机采样. 定义抽象算法Samp为以给定种子从分布中采样随机变量的过程:

$$x \leftarrow \mathsf{Samp}(D; \mathsf{seed}),$$

其中D为分布, seed是用于采样x的随机种子. 对于空种子$\mathsf{seed} = \epsilon$, 该过程与$x \xleftarrow{\$} D$ 相同且完全随机. 否则x的采样对于相同的种子seed总是确定的.

以

$$(x_1, x_2, \cdots, x_t) \leftarrow \mathsf{Samp}(D_1, D_2, \cdots, D_t; \mathsf{seed})$$

表示按分布D_i采样随机变量x_i的过程, 其中$1 \le i \le t$.

3 设计原理

LAC的设计考虑如下准则:

- 安全性:
  - 基于环LWE的可证明安全性;
  - 能抵抗所有已知攻击.
- 效率:
  - 高速计算;
  - 密钥和密文规模小.
- 正确性: 解密错误率低;
- 灵活性: 易于为不同安全级别类型设置参数.

通常, 环LWE的困难性主要由错误率α和维度n决定, 其中$\alpha = \frac{\sigma}{q}\sqrt{2\pi}$是噪音量级(用标准差$\sigma$度量)和模数$q$的比值, 因此噪音分布和模数的选择十分关键. 根据[3,5,2]中的分析, 合适的维度n为$2^9 = 512$和$2^{10} = 1024$. 在基于模LWE的方案中也使用$2^8 = 256$. 对于这些n的选择, 为了使用数论转换(NTT) 技术来提升多项式乘法的效率, 许多基于环LWE的密码方案中选择模数q为12289或7681[14,5,8]. 实际上, 对模数q的这些选择并不是出于安全性的考虑.

为了兼顾方案安全性和参数紧凑性, 我们的主要设计理念是选择尽可能小的模数q来实现带宽效率. 因此, LAC考虑字节规模的模数, 例如$q = 251$. 同时, 选择较窄的噪音分布, 使α的数值维持在合理范围. 尽管我们不能使用NTT来加速多项式乘法, 由于模数仅为字节规模, 我们可以转而使用Intel的AVX2指令集, 在一个指令周期内处理多个乘法运算, 从而提升计算效率.

为了简化秘密和错误的采样, 我们的基本思路是使用$\{-1, 0, 1\}$上的中心二项分布, 然而此时解密错误率会非常大. 为了解决这一问题, 我们使用大分组大码距的纠错码来降低错误率, 例如二元BCH码. 原则上, 任何具有足够纠错能力的纠错码都可以用在LAC方案中, 例如Goppa码, LDPC 码等. 为了避免高汉明重量攻击, 我们也使用固定汉明重量的n重中心二项分布.

对于使用困难学习问题的算法, 为了实现不同的安全级别, 最常见的方式是使用不同的维度n. 但是对于环$R_q = \mathbb{Z}_q[x]/(x^n + 1)$, 当$n$为2的方幂时, 可选择的$n$很少, 例如$n = 512$或$n = 1024$. 为了提供不同的安全强度级别, 我们使用$\{-1, 0, 1\}$上不同标准差的中心二项分布结合维度$n$的选择来实现安全性和密文大小的折衷.

基于具有IND-CPA安全性的基础公钥加密方案, 很容易得到临时密钥交换方案. 我们使用FO转换的变形[17,18,19,20]来得到IND-CCA安全的密钥封装机制, 并使用FSXY转换[15,16]来得到认证密钥交换协议.

4 算法描述

本部分将给出LAC.KEX密码系统的描述, 包括基础密钥交换算法LAC.KEX和认证密钥交换算法LAC.AKE. 为了完整性和评估方便，本文档包含了LAC.PKE的所有内容。

4.1 LAC.PKE

IND-CPA安全的公钥加密方案LAC.PKE是整个LAC密码系统的基础. 其中有三个算法:

- 密钥生成算法KG, 如算法1中所示.
- 加密算法Enc, 如算法2中所示.
- 解密算法Dec, 如算法3中所示.

符号. 令q为模数, 定义多项式环$R_q = \mathbb{Z}_q/(x^n + 1)$.

定义消息空间$\mathcal{M}$为$\{0, 1\}^{l_m}$, 随机种子空间为$\mathcal{S}$为$\{0, 1\}^{l_s}$, 其中l_m, l_s为正整数, 将在随后指定.

算法中使用n重独立同分布的中心二项分布Ψ_σ^n. 此外, 我们还使用固定汉明重量的n重中心二项分布$\Psi_\sigma^{n,h}$. 其中参数的具体值将在随后给出.

子程序. 令ECCEnc, ECCDec为纠错机制的编码和解码子程序, 用于进行消息$\boldsymbol{m} \in \{0, 1\}^{l_m}$及其编码$\widehat{\boldsymbol{m}} \in \{0, 1\}^{l_v}$之间的变换, 其中正整数$l_v$表示编码的长度, 根据具体的参数设置而定.

算法描述.

算法LAC.PKE.KG随机生成一对公私钥(pk, sk).

Algorithm 1 LAC.PKE.KG()

Ensure: 输出一对公私钥(pk, sk).

1: $\mathsf{seed}_{\boldsymbol{a}} \xleftarrow{\$} \mathcal{S}$
2: $\boldsymbol{a} \leftarrow \mathsf{Samp}(U(R_q); \mathsf{seed}_{\boldsymbol{a}}) \in R_q$
3: $\boldsymbol{s} \xleftarrow{\$} \Psi_{\sigma}^{n,h}$
4: $\boldsymbol{e} \xleftarrow{\$} \Psi_{\sigma}^{n,h}$
5: $\boldsymbol{b} \leftarrow \boldsymbol{as} + \boldsymbol{e} \in R_q$
6: **return** $(pk := (\mathsf{seed}_{\boldsymbol{a}}, \boldsymbol{b}), sk := \boldsymbol{s})$

算法LAC.PKE.Enc输入pk和消息$\boldsymbol{m}$, 以随机数seed加密$\boldsymbol{m}$. 其中子程序ECCEnc将消息$\boldsymbol{m}$编码为$\widehat{\boldsymbol{m}}$. 当seed未指定时, 算法是随机的. 否则, 对于相同的seed, 算法是确定的.

Algorithm 2 LAC.PKE.Enc$(pk = (\mathsf{seed}_{\boldsymbol{a}}, \boldsymbol{b}), \boldsymbol{m} \in \mathcal{M}; \mathsf{seed} \in \mathcal{S})$

Ensure: 输出密文$\boldsymbol{c}$.

1: $\boldsymbol{a} \leftarrow \mathsf{Samp}(U(R_q); \mathsf{seed}_{\boldsymbol{a}}) \in R_q$
2: $\widehat{\boldsymbol{m}} \leftarrow \mathsf{ECCEnc}(\boldsymbol{m}) \in \{0,1\}^{l_v}$
3: $(\boldsymbol{r}, \boldsymbol{e}_1, \boldsymbol{e}_2) \leftarrow \mathsf{Samp}(\Psi_{\sigma}^{n,h}, \Psi_{\sigma}^{n,h}, \Psi_{\sigma}^{l_v}; \mathsf{seed})$
4: $\boldsymbol{c}_1 \leftarrow \boldsymbol{ar} + \boldsymbol{e}_1 \in R_q$
5: $\boldsymbol{c}_2 \leftarrow (\boldsymbol{br})_{l_v} + \boldsymbol{e}_2 + \lfloor \frac{q}{2} \rceil \cdot \widehat{\boldsymbol{m}} \in \mathbb{Z}_q^{l_v}$
6: **return** $\boldsymbol{c} := (\boldsymbol{c}_1, \boldsymbol{c}_2) \in R_q \times \mathbb{Z}_q^{l_v}$

算法LAC.PKE.Dec输入sk和密文$\boldsymbol{c}$, 恢复对应的明文$\boldsymbol{m}$. 其中子程序ECCDec输入编码$\widehat{\boldsymbol{m}}$,将其中的码字解码. 通常, 解码将得到消息$\boldsymbol{m} \in \mathcal{M}$. 当出现解码错误时, 所返回的消息$\boldsymbol{m}$不在$\mathcal{M}$ 中.

Algorithm 3 LAC.PKE.Dec($sk = \boldsymbol{s}, \boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$)

Ensure: 输出明文$\boldsymbol{m}$.

$\boldsymbol{u} \leftarrow \boldsymbol{c}_1\boldsymbol{s} \in R_q$
$\widetilde{\boldsymbol{m}} \leftarrow \boldsymbol{c}_2 - (\boldsymbol{u})_{l_v} \in \mathbb{Z}_q^{l_v}$
for $i = 0$ to $l_v - 1$ **do**
  if $\frac{q}{4} \leq \widetilde{\boldsymbol{m}}_i < \frac{3q}{4}$ **then**
    $\widehat{\boldsymbol{m}}_i \leftarrow 1$
  else
    $\widehat{\boldsymbol{m}}_i \leftarrow 0$
  end if
end for
$\boldsymbol{m} \leftarrow \mathsf{ECCDec}(\widehat{\boldsymbol{m}})$
return $\boldsymbol{m}$

4.2 LAC.KEM

IND-CCA安全的密钥封装机制LAC.KEM是对IND-CPA安全的公钥加密方案LAC.PKE应用Fujisaki-Okamoto转换而来[17,19]. 这一方法在[27]中建议, 并在多个方案中使用, 如[8].

LAC.KEM包含如下三个算法:

- 密钥生成算法KG, 与LAC.PKE的密钥生成算法相同, 如算法1中所示.
- 封装算法Enc, 如算法4中所示.
- 解封装算法Dec, 如算法5中所示.

符号. LAC.KEM的描述中所用的记号与LAC.PKE中相同. 此外, 我们还使用hash函数$G : \{0,1\}^{l_m} \rightarrow \mathcal{S} \in \{0,1\}^{l_s}$和$H : \{0,1\}^* \rightarrow \{0,1\}^{l_k}$, 用来进行FO转换和生成封装密钥, 其中$l_k$表示会话密钥的长度, 可以随安全级别而变化. 在LAC中我们总是设置$l_k = l_m$. G和H的具体选择将在第7部分说明.

算法LAC.KEM.Enc输入pk和随机种子$\mathsf{seed}_{\boldsymbol{m}}$，基于该种子生成随机消息$\boldsymbol{m}$, 调用LAC.PKE.ENC 加密$\boldsymbol{m}$. 若输入未提供随机种子$\mathsf{seed}_{\boldsymbol{m}}$，则首先生成该随机种子。

Algorithm 4 LAC.KEM.Enc($pk; \mathsf{seed}_{\boldsymbol{m}}$)

Ensure: 输出一对密文和封装密钥$(\boldsymbol{c}, K)$.

$\boldsymbol{m} \leftarrow \mathsf{Samp}(U(\mathcal{M}), \mathsf{seed}_{\boldsymbol{m}}) \in \mathcal{M}$
$\mathsf{seed} \leftarrow G(\boldsymbol{m}|pk) \in \mathcal{S}$
$\boldsymbol{c} \leftarrow \mathsf{LAC.PKE.Enc}(pk, \boldsymbol{m}; \mathsf{seed})$
$K \leftarrow H(\boldsymbol{m}) \in \{0,1\}^{l_k}$
return $(\boldsymbol{c}, K)$

解封装算法LAC.KEM.Dec输入sk和密文, 通过调用LAC.PKE.Dec恢复消息. 随后其通过重加密来验证解密的正确性. 当验证通过时，其返回封装密钥. 否则, 其使用私钥和密文生成一个伪随机的会话密钥.

Algorithm 5 LAC.KEM.Dec$(sk, \boldsymbol{c})$

Ensure: 输出封装密钥K.

$\boldsymbol{m} \leftarrow \mathsf{LAC.PKE.Dec}(sk, \boldsymbol{c})$
$K \leftarrow H(\boldsymbol{m})$
$\mathsf{seed} \leftarrow G(\boldsymbol{m}|pk) \in \mathcal{S}$
$\boldsymbol{c}' \leftarrow \mathsf{LAC.PKE.Enc}(pk, \boldsymbol{m}; \mathsf{seed})$
if $\boldsymbol{c}' \neq \boldsymbol{c}$ **then**
    $K \leftarrow H(H(sk), c)$
end if
return K

4.3 LAC.KEX

从IND-CPA安全的加密方案LAC.PKE可直接得到被动安全的无认证密钥交换协议LAC.KEX, 如图2中所示.

记号. LAC.KEX的描述中所用记号与LAC.PKE中相同. 此外, 用hash函数$H : \{0,1\}^* \rightarrow \{0,1\}^{l_k}$来生成会话密钥, 其中$l_k$表示会话密钥的长度, 可以随安全级别而变化.

图 2. LAC.KEX: 基于格的无认证密钥交换协议

参数: LAC.PKE的说明
$H : \{0,1\}^* \rightarrow \{0,1\}^{l_k}$

Alice		Bob
$(pk, sk) \xleftarrow{\$} \mathsf{LAC.PKE.KG}()$	$\xrightarrow{pk}$	
		$r \xleftarrow{\$} \{0,1\}^{l_m}$
	$\xleftarrow{c}$	$c \xleftarrow{\$} \mathsf{LAC.PKE.Enc}(pk, r)$
$r \leftarrow \mathsf{LAC.PKE.Dec}(sk, c)$		$K \leftarrow H(pk, r) \in \{0,1\}^{l_k}$
$K \leftarrow H(pk, r) \in \{0,1\}^{l_k}$		

此外, 我们也可以直接从LAC.KEM构造被动安全的密钥交换协议.

4.4 LAC.AKE

使用[15,16]中的通用转换框架, 可以基于IND-CPA安全的公钥加密方案LAC.PKE和IND-CCA安全的密钥封装机制LAC.KEM 构造认证密钥交换协议LAC.AKE. LAC.AKE在Canetti-

Krawczyk模型中安全, 具有弱前向安全性[10], 可抗密钥泄露冒充攻击(KCI), 以及最大泄露攻击(MEX)[15,16]. 协议描述如图3所示.

图 3. LAC.AKE: 基于格的认证密钥交换协议

参数: LAC.KEM和LAC.PKE的说明
$G : \{0,1\}^* \to \{0,1\}^{l_s}$, $H : \{0,1\}^* \to \{0,1\}^{l_k}$

Alice		Bob
$(pk_A, sk_A) \xleftarrow{\$} \mathsf{LAC.KEM.KG}()$		$(pk_B, sk_B) \xleftarrow{\$} \mathsf{LAC.KEM.KG}()$
静态公钥: pk_A		静态公钥: pk_B
静态私钥: sk_A		静态私钥: sk_B
$(pk, sk) \xleftarrow{\$} \mathsf{LAC.PKE.KG}()$		
$r_1 \xleftarrow{\$} \{0,1\}^{l_s}$		
$\mathsf{seed}_1 \leftarrow G(r_1, sk_A)$		
$(\boldsymbol{c}_1, K_1) \leftarrow \mathsf{LAC.KEM.Enc}(pk_B; \mathsf{seed}_1)$	$\xrightarrow{pk, \boldsymbol{c}_1}$	
		$K_1 \leftarrow \mathsf{LAC.KEM.Dec}(sk_B, \boldsymbol{c}_1)$
		$r_2 \xleftarrow{\$} \{0,1\}^{l_s}$
		$\mathsf{seed}_2 \leftarrow G(r_2, sk_B)$
		$(\boldsymbol{c}_2, K_2) \leftarrow \mathsf{LAC.KEM.Enc}(pk_A; \mathsf{seed}_2)$
		$K_3 \xleftarrow{\$} \{0,1\}^{l_m}$
	$\xleftarrow{\boldsymbol{c}_2, \boldsymbol{c}_3}$	$\boldsymbol{c}_3 \xleftarrow{\$} \mathsf{LAC.PKE.Enc}(pk, K_3; \epsilon)$
$K_2 \leftarrow \mathsf{LAC.KEM.Dec}(sk_A, \boldsymbol{c}_2)$		
$K_3 \leftarrow \mathsf{LAC.PKE.Dec}(sk, \boldsymbol{c}_3)$		
$K \leftarrow H(pk_A, pk_B, pk, \boldsymbol{c}_3, K_1, K_2, K_3)$		$K \leftarrow H(pk_A, pk_B, pk, \boldsymbol{c}_3, K_1, K_2, K_3)$

5 参数选择

几乎所有的基于格的公钥加密和密钥交换方案, 除了基于NTRU的几类, 都遵循类似的框架. 为了保证其可证明安全, 有许多关于环, 模数, 错误等的选择理论. 然而, 这些理论并未对具体的参数选择给出指导. 因此, 为基于(环)LWE的方案选择具体参数成为后续研究中的主要问题, 也是许多方案的主要区别. 本部分将介绍LAC的具体参数选择.

5.1 模数

根据LAC的设计原理, 降低模数是主要目标. 如此前所述, 密钥和密文大小主要由维度和模数决定. 使用2次幂分圆环时, 维度n的选择十分有限, 因此我们主要依靠小模数来降低负载大小. 但是模数也不能过小, 它需要足够大, 以容纳$\sqrt{2n}$量级的噪音. 在LAC中, 我们选择字节级模数. 字节是大多数处理器的基本运算单元. 这样的选择使公钥和密文紧凑, 也利于实现. 其缺点在于当模数变小时, 解密错误率将上升.

在设计中, 我们考虑了三种字节级模数:

– 2的方幂, 即$q = 256$.
– 素数$q \equiv 1 \mod 2^k$, 即$q = 257$.
– 占一个字节的最大素数, 即$q = 251$.

最终, 出于安全性和简化代数结构的考虑, 我们选择了$q = 251$. 同时, 从实现的角度, $q = 256, 257$可能带来更好的效率, 这也是我们未来需要探索的问题.

5.2 秘密和错误分布

环LWE问题的秘密和错误分布的选择主要遵循两个原则. 首先, 秘密和错误需要足够大以保证环LWE问题的困难性. 其次, 秘密和错误需要足够小以保证解密算法的正确性. 常见选择有离散高斯分布和中心二项分布. 高斯分布的采样需要消耗大量熵, 并且难以在常数时间内实现; 当其使用查表实现时, 对基于存储的侧信道攻击也较脆弱. 因此, 我们使用中心二项分布.

在实现中, 标准差为$\sqrt{\lambda/2}$的中心二项分布的生成方式为计算$\Sigma_{i=1}^{\lambda}(b_i - \hat{b}_i)$, 其中$b_i, \hat{b}_i \in \{0, 1\}$是均匀随机生成的比特. 由于环LWE的困难性主要由维度n和错误-模数比率决定, 当我们使用字节级模数时, 即使对于很小的错误分布, 错误- 模数比率也变得足够大, 这使得我们可以使用最简单的中心二项分布, 即选择$\lambda = 1$, 错误向量的每个元素都通过$b - \hat{b}$生成, 其中$b, \hat{b}$是均匀随机的比特. 这样得到的中心二项分布即此前所述的Ψ_1. 此外, 我们还使用更小的中心二项分布$\Psi_{\frac{1}{2}}$.

1. $\Psi_1 : \Pr[x = 0] = 1/2, \Pr[x = \pm 1] = 1/4$.
2. $\Psi_{\frac{1}{2}}$: $\Pr[x = 0] = 3/4, \Pr[x = \pm 1] = 1/8$.

在LAC中, 为了避免高汉明重量攻击, 我们对于错误和秘密向量$\boldsymbol{s}, \boldsymbol{e}, \boldsymbol{r}, \boldsymbol{e}_1$使用固定汉明重量的$n$重中心二项分布$\Psi_{\sigma}^{n,h}$, 对错误向量$\boldsymbol{e}_2$使用独立同分布的$n$重中心二项分布$\Psi_{\sigma}^{n}$. 其原因在于$\boldsymbol{e}_2$对高汉明重量攻击没有影响, 因此我们对$\boldsymbol{e}_2$使用采样效率更高的独立同分布的$n$重中心二项分布.

5.3 解密错误分析

如解密算法中所示, 消息的恢复有两个步骤. 首先, 从密文中恢复纠错码的码字$\widehat{\boldsymbol{m}}$. 随后, 从码字中恢复消息$\boldsymbol{m}$. 显然, 有

$$\begin{aligned}\widetilde{\boldsymbol{m}} &= \boldsymbol{c}_2 - (\boldsymbol{c}_1\boldsymbol{s})_{l_v} \\ &= (\boldsymbol{b}\boldsymbol{r})_{l_v} + \boldsymbol{e}_2 + \lfloor \tfrac{q}{2} \rceil \widehat{\boldsymbol{m}} - (\boldsymbol{c}_1\boldsymbol{s})_{l_v} \\ &= ((\boldsymbol{a}\boldsymbol{s} + \boldsymbol{e})\boldsymbol{r})_{l_v} + \boldsymbol{e}_2 + \lfloor \tfrac{q}{2} \rceil \widehat{\boldsymbol{m}} - ((\boldsymbol{a}\boldsymbol{r} + \boldsymbol{e_1})\boldsymbol{s})_{l_v} \\ &= (\boldsymbol{e}\boldsymbol{r} - \boldsymbol{e_1}\boldsymbol{s})_{l_v} + \boldsymbol{e}_2 + \lfloor \tfrac{q}{2} \rceil \widehat{\boldsymbol{m}}.\end{aligned} \tag{1}$$

令$\boldsymbol{w} = (\boldsymbol{e}\boldsymbol{r} - \boldsymbol{e_1}\boldsymbol{s})_{l_v} + \boldsymbol{e}_2$, 则每个$\tilde{m}_i$的错误率为$\delta = 1 - \Pr[-\lfloor \frac{q}{4} \rceil < w_i < \lfloor \frac{q}{4} \rceil]$. 若$\boldsymbol{s}, \boldsymbol{e}, \boldsymbol{r}, \boldsymbol{e}_1, \boldsymbol{e}_2$是从标准差为$\sigma$, 均值为0的小分布中随机选取的, 则根据中心极限定理, w_i的分布非常接近均值为0, 标准差为$\sigma^2\sqrt{2n}$的高斯分布. 因此, 每个比特的错误率可以

用高斯错误函数近似为$\delta \approx 1 - \mathsf{erf}(\frac{\lfloor q/4 \rceil}{\sqrt{2}(\sigma^2\sqrt{2n})})$. 例如, 对于$n = 512, q = 251$, 分布$\Psi_1$, 标准差为$\sigma = 1/\sqrt{2}$, 则每个比特的错误率估计为

$$\delta \approx 1 - \mathsf{erf}(\frac{\lfloor 251/4 \rceil}{\sqrt{2}((1/\sqrt{2})^2\sqrt{2 \times 512})}) \approx 2^{-13.195}.$$

假设BCH码最多可以纠l_t个错误, 且码字长度为$l_n = l_v$, 假设$\boldsymbol{w}$的系数彼此不相关, 我们可以计算对于消息$\boldsymbol{m}$, 其解密错误率为

$$\Delta \approx \sum_{j=l_t+1}^{l_v} \left(\binom{l_v}{j} \delta^j (1-\delta)^{l_v - j} \right) \tag{2}$$

以上估算中对于n重中心二项分布使用的是独立同分布的模型, 我们所使用的固定汉明重量的n重中心二项分布中每个样本的汉明重量都是独立选取时的期望值, 故以上的分析仍然适用.

5.4 纠错码

LAC的字节级模数带来了非常高的解密错误率. 使用轻量级的纠错方法例如D2或D4不能处理如此高的错误率. 因此, 我们需要使用较强的纠错方法. 在编码领域有许多强纠错码, 例如BCH, Goppa, LDPC, Turbo和Polar. 原则上, 任意具有足够纠错能力的编码都可以用在LAC中. 出于简洁和效率的考虑, 我们在实现中选择了BCH码.

对于码长为l_n, 设计距离$l_d < l_n$的BCH码, 我们使用SageMath中的"codes.BCHCode($GF(2), l_n, l_d$)"来选择具体参数.

5.5 推荐参数级别

我们推荐的参数级别设置如表1中所示:

安全级别	n	q	**dis**	**ecc**	$l_{\boldsymbol{m}}$	**pk**	**sk**	**ct**	**bit-er**	**dec-er**
LAC-test	512	251	Ψ_{512}^{128}	BCH[511,256,7]	256	544	512	656	$2^{-42.24}$	2^{-140}
LAC-light	512	251	Ψ_{512}^{128}	双向奇偶校验码+D2	128	544	512	664	$2^{-81.73}$	2^{-150}
LAC-128	512	251	Ψ_{512}^{256}	BCH[255, 128, 17]+D2	128	544	512	704	$2^{-22.26}$	2^{-151}
LAC-192	1024	251	Ψ_{1024}^{256}	BCH[511, 256, 17]+D2	256	1056	1024	1352	$2^{-42.24}$	2^{-324}
LAC-256	1024	251	Ψ_{1024}^{384}	BCH[511, 256, 41]+D2	256	1056	1024	1448	$2^{-20.01}$	2^{-303}

dis 秘密和噪音分布　　**ecc** 纠错码
$l_{\boldsymbol{m}}$ 消息长度　　**sk** 私钥大小
pk 公钥大小　　**ct** 密文大小
bit-er 不用纠错码时的单比特错误率　　**dec-er** 解密错误率

表 1. LAC.PKE的推荐参数

与第一轮参数相比，我们给出了新的LAC-light参数，其单比特错误率较低，我们采用了简单的双向奇偶校验码纠正1个比特错误，不再依赖BCH编码进行纠错，故实现了更高的效率，其具体安全性评估只比LAC-128 略低，具体见6.3. 同时，为了将解密错误率降至2^{-l}以下，我们调整了三个安全级别参数的纠错码参数配置。

为了方便评估团队的测试，我们增加了一套符合竞赛要求的指定参数LAC-test, 作为测试使用。其基本参数与LAC-light相同，仅仅是增大了消息空间以满足竞赛指定参数的要求，并相应调整了纠错码参数以满足消息空间的需求。

由于在实际中, 公钥加密方案主要用来传输对称加密方案使用的会话密钥, 根据安全级别设置消息长度便已足够, 因此我们将128比特安全级别的消息长度选择为128, 256比特安全级别的消息空间选择为256. 由于我们的192比特安全级别参数的实际评估安全性远高于192, 其量子核心SVP算法评估的复杂性为259, 因此该参数可以用于256比特安全级别, 因而我们将其消息空间定位256.

BCH码的参数选择是为了达到合适的解密错误率, 同时保持高性能. 首先, 我们设置l_m与消息长度相同. 对于$l_m = 128$, 最小可用的BCH 码长$l_n = 255$, 对于$l_m = 256$, 最小可用的BCH 码长$l_n = 511$. 最后, 对于LAC-128和LAC-256, 我们选择$l_d = 17$, 可以纠最多8比特的错误. 对于LAC-256, 我们选择$l_d = 41$, 可以纠最多20比特的错误. 为了在提升纠错能力的同时保持计算性能受到较小的影响，我们使用了D2算法配合BCH进行纠错.

每个系数的错误率是通过所有错误项的卷积估计的, 为了最小化密文的大小, 在实现中, 我们丢弃$\boldsymbol{c}_2$的每个系数的低4比特. 这带来了额外的$[-7, 7]$ 上的均匀随机(在环LWE假设下)的错误.

公钥包括32字节的种子$\mathsf{seed}_{\boldsymbol{a}}$, 以及$n$字节向量$\boldsymbol{b}$. 私钥是$n$字节向量. 也可以只存储32字节的种子以生成私钥, 从而最小化存储, 代价是略影响解密速度.. 当使用Fujisaki-Okamoto转换来得到CCA安全性时, 私钥中也包括对应的公钥, 使得解密算法可以重加密来检查密文的有效性. 因此在LAC的CCA安全的密钥封装机制中私钥大小为$2n + 32$字节.

最后, 密文包括n字节向量$\boldsymbol{c}_1$, 以及来自$\boldsymbol{c}_2$的l_v字节. 对于LAC-light参数集, $l_v = l_m + 3 \times 8$, 其中3是纠错码冗余数据大小; 对于 LAC-128参数集, $l_v = l_m + 8 \times 8$, 其中8是纠错码冗余数据大小; 对于LAC-192参数集, $l_v = l_m + 9 \times 8$, 其中9 是纠错码冗余数据的大小; 对于LAC-256, $l_v = (l_m + 21 \times 8)$.

6 安全性分析

6.1 具体安全性

关于LAC的具体安全性分析, 我们考虑环LWE在选定参数下已知最好的通用攻击. 这些攻击将环LWE问题视为普通的LWE问题. 这些攻击为大家熟知, 其攻击成本易于理解. 此外, 我们也考虑针对LAC的特殊设计的专用攻击, 即子域攻击和高汉明重量攻击.

6.2 通用攻击

存在许多解决LWE问题的通用算法, [3,29]中给出了这些已知技术的综述. 其中使用BKZ算法的格基约化攻击[11]比穷搜, 组合和代数攻击算法效果更好. 简单起见, 类似于[4]中的分析, 我们主要关注两类嵌入攻击, 即通常所说的原始攻击和对偶攻击. 表2中给出了对这些攻击的安全性估计.

算法	原始攻击			对偶攻击		
	经典	量子	分块大小	经典	量子	分块大小
LAC-test	131	118	448	130	118	445
LAC-light	131	118	448	130	118	445
LAC-128	148	135	509	147	133	505
LAC-192	288	261	986	286	259	978
LAC-256	308	279	1054	305	277	1044

经典: 经典复杂度　　**量子:** 量子复杂度

表 2. LAC的具体安全性

原始攻击.

原始攻击中, 首先从LWE样本中构造一个具有uSVP实例的格; 随后, 使用BKZ算法恢复该唯一最短向量. 简单地说, 给定LWE实例$(\boldsymbol{A}, \boldsymbol{b} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e})$, $\boldsymbol{A} \in \mathbb{Z}_q^{m\times n}$, 目标格维度为

$$\Lambda_{\boldsymbol{A}} = \{\boldsymbol{x} \in \mathbb{Z}^{m+n+1} : (\boldsymbol{A}|\boldsymbol{I}_m| - \boldsymbol{b})\boldsymbol{x} = \boldsymbol{0} \mod q\}.$$

容易验证, 当$\boldsymbol{s}$和$\boldsymbol{e}$足够短时, $\boldsymbol{v} = (\boldsymbol{s}, \boldsymbol{e}, 1)$是uSVP 的解. 例如, 如[4]中所示, 攻击成功当且仅当$\sigma\sqrt{b} \leq \delta^{2b-d-1} \times q^{m/d}$, 其中$\sigma$是错误和秘密分布的标准差, $\delta = ((\pi b)^{1/b} b/2\pi e)^{1/(2(b-1))}$.

BKZ算法通过调用一个子程序多项式次来渐进地处理格基, 从而解决维度(即分块大小)为b的子格的精确最短向量问题, 例如调用(量子)筛法. 该方法即BKZ-core-(Q)Sieving, 其复杂度只依赖于BKZ算法寻找唯一解时需要的分块维度b. 根据[4], SVP谕言对经典筛法的最佳复杂度为$\sqrt{3/2}^{b+o(b)} \approx 2^{0.292b}$, 对量子筛法为$\sqrt{13/9}^{b+o(b)} \approx 2^{0.265b}$.

对偶攻击.

对偶攻击中, 首先构造此前所述的原始格的对偶格, 随后使用对偶格来解决判定LWE问题. 给定LWE实例$(\boldsymbol{A}, \boldsymbol{b} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e})$, $\boldsymbol{A} \in \mathbb{Z}_q^{m\times n}$, 构造维度为$d = m + n$的目标格

$$\Lambda_{\boldsymbol{A}}^{\perp} = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z}^m \times \mathbb{Z}^n : \boldsymbol{A}^t\boldsymbol{x} = \boldsymbol{y} \mod q\}.$$

同样, [4]中指出, BKZ可以找到向量$\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{y})$ of length $l = \delta^{d-1} q^{n/d}$, 其中$\boldsymbol{v}^t\boldsymbol{b}$和均匀分布之间的距离界为$\epsilon = 4\exp(-2\pi^2\tau^2)$, $\tau = l\sigma/q$. 这以优势ϵ攻破判定LWE问题.

类似于原始攻击, 对偶攻击的具体安全性同样依赖于BKZ算法的复杂度. 当使用BKZ-core-(Q)Sieving时需注意: 攻击者可以将筛法重复$R = max(1, 1/(\gamma\epsilon^2))$次, 从而将$\epsilon$放大到1/2. 这个操作对攻击者来说几乎无成本, 因为筛法将产生$\gamma = 2^{0.2075b}$个向量, 远超重复所需的短向量数$1/\epsilon^2$.

安全性估计. 我们使用带核心(量子)筛法模型的BKZ模拟器来估计方案的安全性. BKZ算法所需的分块大小以及对应的安全性估计如表2中所示.

6.3 专用攻击

需要强调的是此处分析的两种攻击的表现并未强于一般攻击.

子域攻击. 子域攻击已存在多年[6,1,7,21], 用其分析LAC的想法最初由Alperin-Sheriff提出[26]. 当模数$q = 251$时$x^n + 1$有两个因子:

$$x^n + 1 = (x^{n/2} + 91x^{n/4} + 250)(x^{n/2} + 160x^{n/4} + 250).$$

换而言之, 存在由多项式$\boldsymbol{g}$和$\boldsymbol{h}$定义的两个子域, 其中$\boldsymbol{g} = x^{n/2} + 91x^{n/4} + 250$且$\boldsymbol{h} = x^{n/2} + 160x^{n/4} + 250$.

给定$(\boldsymbol{a}, \boldsymbol{b} = \boldsymbol{as} + \boldsymbol{e})$, 通过在子域上检查样本, 可以恢复$(\boldsymbol{s}, \boldsymbol{e})$. 从$(\boldsymbol{a} \bmod \boldsymbol{g}, \boldsymbol{b} \bmod \boldsymbol{g})$ 足以恢复$(\boldsymbol{s}_g := \boldsymbol{s} \bmod \boldsymbol{g}, \boldsymbol{e}_g := \boldsymbol{e} \bmod \boldsymbol{g})$, 对$(\boldsymbol{s}_h, \boldsymbol{e}_h)$ 也类似. 之后可以使用中国剩余定理直接恢复$(\boldsymbol{s}, \boldsymbol{e})$.

分析. 该攻击的关键点在于, 转换到子域之后格的维度减半, 因此, BKZ复杂度对新的子格可能降低. 我们的分析显示, 子域中对应的向量, $(\boldsymbol{s}_g, \boldsymbol{e}_g)$, 将大于根据高斯启发式估计的最短向量长度. 换而言之, 即使可以在维度减半的格上进行格基约化攻击, 也无法恢复目标向量. 虽然转换到子域降低了维度, 同时也增加了$(\boldsymbol{s}_g, \boldsymbol{e}_g)$的大小($(\boldsymbol{s}_h, \boldsymbol{e}_h)$类似). 具体而言, $(\boldsymbol{s}, \boldsymbol{e})$ 是系数在$\{-1, 0, 1\}$中选取的小多项式, 而$(\boldsymbol{s}_g, \boldsymbol{e}_g)$的系数将分布在$\{0, \pm1, \pm2, \pm91\}$中. Alperin-Sheriff 指出, 将$\boldsymbol{s}$ and $\boldsymbol{e}$乘以11, $(\boldsymbol{s}_g, \boldsymbol{e}_g)$的所有系数将分布在$[-25, 25]$区间.

令$\boldsymbol{A} = [\boldsymbol{A}_g|\boldsymbol{I}|11 \times \boldsymbol{b}_g]$, 其中$\boldsymbol{A_g}$表示$\boldsymbol{a}_g$生成的矩阵, 若$\boldsymbol{z} = [11 \times \boldsymbol{s}_g|11 \times \boldsymbol{e}_g| - 1]$是$\boldsymbol{Az} = 0 \bmod q$的最短解, 则我们可以使用原始攻击恢复$\boldsymbol{z}$. 需注意, 通过子域攻击, 原始攻击的维度从$d = 2n + 1$降为$d = n + 1$. 由于$\boldsymbol{A}$是随机矩阵, q元格$\Lambda_q^{\perp}(\boldsymbol{A})$可视为随机格[12], 因此可以用高斯启发式来估计该格中的最短向量的长度:

$$\lambda_1(\Lambda_q^{\perp}) \approx q^{m/d}\sqrt{\frac{d}{2\pi e}}.$$

在$n = 512$和$n = 1024$时, 最短向量的长度估计分别为86.36和122.4.

另一方面, 我们还需要估计$\boldsymbol{z}$的长度. 根据中心极限定理, $\boldsymbol{z}$的长度近似服从离散高斯分布. 按照我们的实现, $\boldsymbol{z}$对LAC-128近似服从均值方差对为$(253.59, 6.9)$的高斯分布, 对LAC-192为$(253.26, 6.29)$, 对LAC-256v2为$(358.42, 6.86)$[1].

显然, 除可忽略的概率外, $\boldsymbol{z}$的长度比$\boldsymbol{Az} = 0 \bmod q$的解大. 因此$\boldsymbol{z}$并非该格的短向量. 换而言之, 如果要使用子域攻击, 假设敌手可以自由访问子格的SVP谕言, 也无法找到短向量.

总之, 以上描述的子域攻击对LAC 的参数集无影响.

高汉明重量攻击. 高汉明重量攻击是一种选择密文攻击, 由于某些密文中的秘密和错误$(\boldsymbol{r}, \boldsymbol{e}_1)$的汉明重量可能会高于通常情况, 因此可被其利用. D'Anvers等人在文献[13]中描述了对一些后量子方案的高汉明重量攻击尝试. 为了免疫这种攻击, 我们在方案中使用了固定汉明重量的n重中心二项分布. 因此, 目前采用的参数不会受到这种攻击.

7 算法实现和性能

如此前所述, LAC和其它一些基于环LWE的公钥加密方案的主要区别在于我们的参数不支持NTT. 这一部分我们重点介绍为LAC定制的实现方案, 包括通用的多项式乘法, 以及基于AVX2 指令的最优实现.

7.1 多项式乘法

多项式乘法是LAC的实现中最耗时的运算. 在参考实现之外, 我们提供如下两种优化版本:

- **通用优化版本:** 其主要观察是, 由于$\mathbf{s}$和$\mathbf{r}$选择$\{-1, 0, 1\}$, 乘法操作可以用逐比特的`AND` 逻辑运算实现为$a_i \times 1 = a_i$&0xff和$a_i \times 0 = a_i$&0x00. 此外, 我们可以将4项打包进一个`uint64_t`数据类型. 这样, 多项式乘法变为$\mathbf{as} = \sum_{s_i=1} a_i - \sum_{s_i=-1} a_i$. 当$q < 256$时, 理论上可以将8个系数打包进一个`uint64_t`单元. 我们一次处理4个系数, 将剩余空间作为进位缓冲, 从而不用在每次算术操作后执行模运算. 这样效率更好.
- **基于AVX2的版本:** AVX2使我们可以处理256比特的数据类型. 我们可以在单个`__mm256`数据类型中存储32个系数, 并利用`_mm256_maddubs_epi16`指令在一次操作中进行32次乘法和随后的加法运算. 用这一优化可得到约30倍的加速.

7.2 纠错码

纠错码部分使用到的BCH编码, 我们基于linux内核中的BCH编码进行修改,原始代码下载自：https://github.com/jkent/python-bchlib/tree/master/bchlib. 主要的修改是对BCH 解码算法中的循环结构和分支结构进行了常数时间化处理，通过掩码技术固定循环结构的执行次数和控制变量的更新，避免了for, if等语句带来的时间变化。

[1] 该数据是使用SageMath对每个参数集选择多于100,000个随机样本而来. 该实验并非为了给出统计距离的证据; 该均值明显比高斯启发式高出很多.

7.3 基准测试

这一部分提供了LAC.KEX和LAC.AKE性能测试，其中Alice1代表密钥协商发送者计算发送密钥协商数据需要的计算量，Bob代表接收者计算发送数据和提取密钥需要的计算量，Alice2 代表发送者提取协商密钥需要的时间. 测试平台是Win10 操作系统, 运行于Intel i7-770HQ @ 2.8GHz 处理器, 8GB 内存. 具体结果如下：

方案	LAC.KEX (微秒)			LAC.AKE (微秒)		
	Alice1	**Bob**	**Alice2**	**Alice1**	**Bob**	**Alice2**
LAC-test	6.6	13.1	7.6	21.4	43.5	22.5
LAC-light	6.9	12.8	3.9	29.3	40.8	19.6
LAC-128	10.6	19.5	7.8	29.9	65.2	34.3
LAC-192	16.4	30.2	14.7	47.9	105.8	64.6
LAC-256	24.3	43.0	29.8	66.1	158.7	102.3

表 3. LAC.KEX和LAC.AKE方案性能测试-Visual Studio 2019

为了方便对比算法在不同编译器下的性能，我们在相同操作系统和CPU的情况下测试了基于GCC9.1编译器的性能数据, 具体结果如下：

方案	LAC.KEX (微秒)			LAC.AKE (微秒)		
	Alice1	**Bob**	**Alice2**	**Alice1**	**Bob**	**Alice2**
LAC-test	6.2	12.5	4.7	17.2	37.5	20.3
LAC-light	6.2	11.0	4.6	17.2	37.5	17.2
LAC-128	9.4	18.8	9.3	28.2	62.5	34.3
LAC-192	15.6	28.1	15.6	43.8	100.0	57.8
LAC-256	20.3	37.5	26.6	59.3	146.9	93.8

表 4. LAC.KEX和LAC.AKE方案性能测试-GCC9.1

7.4 密钥和密文大小

LAC的密文和密钥大小如下表所示.

安全级别	公钥(字节)	私钥(字节)	密文(字节)
LAC-test	544	1056	656
LAC-light	544	1056	664
LAC-128	544	1056	704
LAC-192	1056	2080	1352
LAC-256	1056	2080	1448

表 5. LAC.PKE及LAC.KEM的密钥密文大小

安全级别	LAC.KEX		LAC.AKE	
	Alice(字节)	Bob(字节)	Alice(字节)	Bob(字节)
LAC-test	544	656	1200	1312
LAC-light	544	664	1208	1328
LAC-128	544	704	1248	1408
LAC-192	1056	1352	2408	2704
LAC-256	1056	1448	2504	2896

表 6. LAC.KEX及LAC.AKE的消息大小

7.5 纠错码的存储成本

定义了BCH纠错码的编码和解码参数的bch_control是LAC中最占存储的部分. 在通用优化实现和AVX2实现中, 这些参数可以在“bch128.h”, “bch192.h” and “bch256.h”中为安全级别分类LAC-128, LAC-192和LAC-256中分别定义. 这些参数的具体成本如下表所示.

安全级别	BCH参数			bch_control(字节)
	代码长度	数据长度	最大错误	
LAC-128	511	256	16	9916
LAC-192	511	256	8	15048
LAC-256	511	256	20	28080

表 7. 纠错码的存储成本

8 优缺点说明

8.1 优点

实现方面:

- LAC可以在支持AVX2指令的英特尔x64处理器上高速实现.
- LAC可以在支持向量指令的ARM处理器如NEON上高速实现.
- LAC的运算在设计上支持并行, 非常适合在多核处理器上实现.

设计的简洁性:

- LAC的设计准则十分简单: 使用字节级模数来减小密文和密钥的尺寸.
- 错误和秘密的分布非常简单, 易于实现.
- LAC的主要运算是多项式乘法, 易于理解.

灵活性:

- LAC提供了三种安全级别, 易于通过维度和错误分布的变化来调节.
- 不同安全级别使用相同的模数.

8.2 缺点

LAC的缺点:

– 需要用强纠错码例如BCH码来保证解密算法的正确性.
– 纠错码实现需要保证常数时间来避免侧信道攻击, 对计算效率有所影响.

参考文献

1. Albrecht, M.R., Bai, S., Ducas, L.: A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In: Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I. pp. 153–178 (2016), https://doi.org/10.1007/978-3-662-53018-4_6
2. Albrecht, M.R., Göpfert, F., Virdia, F., Wunderer, T.: Revisiting the expected cost of solving usvp and applications to LWE. 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. 297–322 (2017), https://doi.org/10.1007/978-3-319-70694-8_11
3. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Mathematical Cryptology 9(3), 169–203 (2015), http://www.degruyter.com/view/j/jmc.2015.9.issue-3/jmc-2015-0016/jmc-2015-0016.xml
4. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Newhope without reconciliation. IACR Cryptology ePrint Archive 2016, 1157 (2016), http://eprint.iacr.org/2016/1157
5. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - A new hope. In: 25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016. pp. 327–343 (2016), https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim
6. Bernstein, D.: A subfield-logarithm attack against ideal lattices (2014), https://blog.cr.yp.to/20140213-ideal.html
7. Biasse, J., Espitau, T., Fouque, P., Gélin, A., Kirchner, P.: Computing generator in cyclotomic integer rings - A subfield algorithm for the principal ideal problem in $l_{|\Delta K|}(\frac{1}{2})$ and application to the cryptanalysis of a FHE scheme. 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. 60–88 (2017), https://doi.org/10.1007/978-3-319-56620-7_3
8. Bos, J.W., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Stehlé, D.: CRYSTALS - kyber: a cca-secure module-lattice-based KEM. IACR Cryptology ePrint Archive 2017, 634 (2017), http://eprint.iacr.org/2017/634
9. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-lwe and security for key dependent messages. In: Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings. pp. 505–524 (2011), https://doi.org/10.1007/978-3-642-22792-9_29
10. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Advances in Cryptology - EUROCRYPT 2001, International Conference on the

Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding. pp. 453–474 (2001), https://doi.org/10.1007/3-540-44987-6_28
11. 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), https://doi.org/10.1007/978-3-642-25385-0_1
12. Daniele Micciancio, O.R.: Lattice-based cryptography. Tech. rep., https://cims.nyu.edu/regev/papers/pqc.pdf (2008)
13. D'Anvers, J., Vercauteren, F., Verbauwhede, I.: On the impact of decryption failures on the security of LWE/LWR based schemes. IACR Cryptology ePrint Archive 2018, 1089 (2018), https://eprint.iacr.org/2018/1089
14. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 40–56 (2013), https://doi.org/10.1007/978-3-642-40041-4_3
15. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly secure authenticated key exchange from factoring, codes, and lattices. In: Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings. pp. 467–484 (2012), https://doi.org/10.1007/978-3-642-30057-8_28
16. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In: 8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13, Hangzhou, China - May 08 - 10, 2013. pp. 83–94 (2013), http://doi.acm.org/10.1145/2484313.2484323
17. Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimum cost. In: Public Key Cryptography, Second International Workshop on Practice and Theory in Public Key Cryptography, PKC '99, Kamakura, Japan, March 1-3, 1999, Proceedings. pp. 53–68 (1999), https://doi.org/10.1007/3-540-49162-7_5
18. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptology 26(1), 80–101 (2013), https://doi.org/10.1007/s00145-011-9114-1
19. 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), https://doi.org/10.1007/978-3-319-70500-2_12
20. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: Ind-cca-secure key encapsulation mechanism in the quantum random oracle model, revisited. In: Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III. pp. 96–125 (2018), https://doi.org/10.1007/978-3-319-96878-0_4
21. 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
22. Lu, X., Liu, Y., Jia, D., Xue, H., He, J., Zhang, Z.: Lac. Tech. rep., https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/LAC.zip (2017)

23. 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. Tech. rep., https://eprint.iacr.org/2018/1009 (2018)
24. 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, French Riviera, May 30 - June 3, 2010. Proceedings. pp. 1–23 (2010), https://doi.org/10.1007/978-3-642-13190-5_1
25. Micciancio, D., Mol, P.: Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In: Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings. pp. 465–484 (2011), https://doi.org/10.1007/978-3-642-22792-9_26
26. NIST: NIST PQC FORUM: LAC, https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/LAC-official-comment.pdf
27. Peikert, C.: Lattice cryptography for the internet. In: Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings. pp. 197–219 (2014), https://doi.org/10.1007/978-3-319-11659-4_12
28. 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), http://doi.acm.org/10.1145/1060590.1060603
29. Schmidt, M., Bindel, N.: Estimation of the hardness of the learning with errors problem with a restricted number of samples. IACR Cryptology ePrint Archive 2017, 140 (2017), http://eprint.iacr.org/2017/140