

## 埃奎斯<sup>1</sup>: 一类基于非对称(M)LWE和(M)SIS的 数字签名和密钥封装机制

Aigis: A famIly of siGnatures and key  
encapsulatIOn mechaniSMS from asymmetric  
(M)LWE and (M)SIS

本文档为密钥封装机制部分

---

<sup>1</sup> 埃奎斯(Aigis, 希腊古典文字): 希腊神话中宙斯和雅典娜使用的盾牌都叫埃奎斯（其中前者又叫宙斯之盾，后者又叫雅典娜之盾），是希腊神话中唯一能抵抗宙斯“雷霆”攻击的法器。

## 基本信息

算法名称： 埃奎斯密钥封装机制 (Aegis-enc)

第一设计者：张江，密码科学技术国家重点实验室

联系电话：15110204521 电子邮箱：jiangzhang09@gmail.com

其他设计者：郁昱，上海交通大学，yyuu@sjtu.edu.cn

范淑琴，密码科学技术国家重点实验室，shuqinfan78@163.com

张振峰，中国科学院软件研究所，zfzhang@tca.iscas.ac.cn

杨糠，密码科学技术国家重点实验室，yangk@sklc.org

算法联系人：杨糠，密码科学技术国家重点实验室

通信地址：北京市海淀区永翔北路9号

联系电话：18510249902 电子邮箱：yangk@sklc.org

提交日期： 2019年2月28日

# 目录

1	引言 .....	1
1.1	埃奎斯密钥封装机制的设计原理 .....	2
1.2	文档结构 .....	5
2	符号及参数 .....	5
2.1	基本定义 .....	7
2.2	基本操作 .....	7
3	埃奎斯密钥封装机制 .....	10
3.1	选择明文攻击安全的公钥加密方案 .....	10
3.2	选择密文攻击安全的密钥封装机制 .....	13
3.3	参数集以及相关函数的实例化 .....	14
4	程序实现及性能 .....	15
4.1	编译和运行程序 .....	15
4.2	测试用例 .....	16
5	可证明安全 .....	22
5.1	困难假设 .....	22
5.2	埃奎斯公钥加密方案的选择明文攻击安全性 .....	25
5.3	埃奎斯密钥封装机制的选择密文攻击安全性 .....	27
6	抵抗已知攻击的能力 .....	27
6.1	原始攻击及其变形 .....	28
6.2	对偶攻击及其变形 .....	31
6.3	埃奎斯密钥封装机制的安全强度 .....	34
7	优缺点 .....	35
A	安全模型 .....	39
A.1	公钥加密方案定义 .....	39
A.2	密钥封装机制定义 .....	40

## 1 引言

当前,格上公钥密码系统的安全性大多是建立在小整数解问题(Small Integer Solutions[2,30], SIS)和带错误的学习问题(Learning with Errors[35], LWE)的困难性之上。简单来说,小整数解问题和带错误的学习问题都与求解模整数方程有关系。令 $n, m, q \in \mathbb{Z}$ 为正整数,  $\alpha, \beta$ 为正实数,  $\chi_\alpha \subseteq \mathbb{Z}$ 是以 $\alpha \in \mathbb{R}$ 为参数的错误分布(通常为高斯分布,或与其相近的二项分布)。无穷范数( $\ell_\infty$ )小整数解问题SIS <sub>$n, m, q, \beta$</sub> 就是给定矩阵 $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ ,计算非零向量 $\mathbf{x} \in \mathbb{Z}^m$ 使其满足 $\mathbf{Ax} = \mathbf{0} \bmod q$ 并且 $\|\mathbf{x}\|_\infty \leq \beta$ ;而对应的计算性带错误的学习问题LWE <sub>$n, m, q, \alpha$</sub> 就是给定样本( $\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e}$ ) $\in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ ,求解 $\mathbf{s} \in \mathbb{Z}_q^n$ ,其中 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}, \mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e} \xleftarrow{\$} \chi_\alpha^m$ 。判定性LWE问题是区分( $\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e}$ )和 $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ 上的均匀随机元素。在一定参数下,判定性LWE问题和计算性LWE问题在多项式时间意义下是等价的[35,28]。此外,SIS问题和LWE问题在一定意义上互为对偶问题。

虽然SIS问题和LWE问题看起来比较简单,但在特定参数下求解这两个问题在平均情况下的复杂度都比求解格上某些问题(例如,最短向量问题)在最坏情况下的复杂度还高[35,30]<sup>1</sup>。由于目前已知的格上困难问题的量子求解算法与传统经典求解算法相比在计算复杂度上并没有本质的降低,以至于大多数国内外研究学者都倾向于相信格上问题是困难的,以及基于格上困难问题设计的密码系统能够抵抗量子计算机攻击。此外,当秘密向量 $\mathbf{s}$ 并不是随机均匀地选自于 $\mathbb{Z}_q^n$ 时,相应LWE的变种问题(称之为正规形LWE问题)也是困难的。特别地,当 $\mathbf{s} \xleftarrow{\$} \chi_\alpha^n$ 与噪音向量 $\mathbf{e}$ 选自于同样的分布时,正规形LWE问题和标准的LWE问题在多项式时间的意义上是等价的[7]。由于正规形LWE问题能够更好的控制噪音增长,因此在文献中被广泛用于设计加密方案[15,12]。

一般来说,SIS问题大多被用于设计数字签名方案,而LWE问题则常常用于设计公钥加密方案。但我们研究发现标准的SIS问题以及LWE问题并不能非常好的满足我们对于密码系统性能,特别是最重要的通信性能的要求。根据对已有格上密码系统设计技术的深入思考和理解,我们提出了非对称的SIS和LWE变形问题。简单来说,非对称SIS问题(Asymmetric SIS, ASIS)ASIS <sub>$n, m_1, m_2, q, \beta_1, \beta_2$</sub> 就是给定矩阵 $\mathbf{A} \in \mathbb{Z}_q^{n \times (m_1+m_2)}$ ,计算非零向量 $\mathbf{x} = (\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{Z}^{m_1+m_2}$ 使其满足 $\mathbf{Ax} = \mathbf{0} \bmod q$ , $\|\mathbf{x}_1\|_\infty \leq \beta_1$ ,并且 $\|\mathbf{x}_2\|_\infty \leq \beta_2$ 。显然,求解ASIS <sub>$n, m_1, m_2, q, \beta_1, \beta_2$</sub> 问题不会比求解SIS <sub>$n, m_1+m_2, q, \min(\beta_1, \beta_2)$</sub> 问题更困难,但同样也不会比求解SIS <sub>$n, m_1+m_2, q, \max(\beta_1, \beta_2)$</sub> 问题更容易。换句话说,在计算困难性上,我们有如下关系

$$\text{SIS}_{n, m_1+m_2, q, \max(\beta_1, \beta_2)} \leq \text{ASIS}_{n, m_1, m_2, q, \beta_1, \beta_2} \leq \text{SIS}_{n, m_1+m_2, q, \min(\beta_1, \beta_2)}$$

---

<sup>1</sup>这种平均困难性到最坏困难性的联系特性实际上是基于格上困难问题的密码方案相对于基于其他困难问题的密码方案独有的优势之一。

成立。这种关系实际上为我们基于ASIS设计密码方案建立了理论基础。此外，我们还提出了一类针对ASIS问题的分析方法，并给出了ASIS问题不同安全强度下的参数选取方法，从而解决了基于ASIS问题密码系统的实际参数选取问题。

相应地，非对称LWE问题（Asymmetric LWE, ALWE） $\text{ALWE}_{n,m,q,\alpha_1,\alpha_2}$ 是给定样本 $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ ，求解 $\mathbf{s} \in \mathbb{Z}_q^n$ ，其中 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$ ,  $\mathbf{s} \xleftarrow{\$} \chi_{\alpha_1}^n$ ,  $\mathbf{e} \xleftarrow{\$} \chi_{\alpha_2}^m$ 。由于求解算法可能会利用秘密向量 $\mathbf{s}$ 的分布信息，我们不能简单地像比较ASIS与SIS一样比较ALWE和LWE的关系，但对于目前已知最好的求解算法我们仍然有关系

$$\text{LWE}_{n,m,q,\min(\alpha_1,\alpha_2)} \leq \text{ALWE}_{n,m,q,\alpha_1,\alpha_2} \leq \text{LWE}_{n,m,q,\max(\alpha_1,\alpha_2)}$$

成立。<sup>2</sup> 更重要的是，文献[22,14,29]研究表明只要 $\chi_{\alpha_1}^n$ 具有足够高的熵（例如， $\{0,1\}^n$ 上的均匀分布），那么我们总可以选择其他参数使得 $\text{ALWE}_{n,m,q,\alpha_1,\alpha_2}$ 达到标准LWE问题的困难强度。这就为我们基于ALWE设计密码系统提供了理论基础。此外，Cheon等人[17]实际上使用了与ALWE相关的变种问题，即 $\mathbf{s}$ 和 $\mathbf{e}$ 选自于不同分布（注意，我们定义的ALWE仅仅指 $\mathbf{s}$ 和 $\mathbf{e}$ 选自于参数不同的相同分布）来设计加密方案。通过综合比较分析和优化最新的LWE求解算法，我们给出了ALWE问题和LWE问题参数之间的近似关系，以及ALWE问题不同安全强度下的参数选取方法，从而解决了基于ALWE问题密码系统的实际参数选取问题。

显然，以上非对称性变种问题的定义可以很自然地推广到环LWE问题/SIS问题(RLWE/RSIS) 和模LWE问题/SIS问题(Module-LWE/SIS, MLWE/MSIS)问题。鉴于众多研究已经表明MLWE/MSIS问题能够在计算效率和通信代价方面实现较好的平衡[19,11]，我们选择使用非对称MLWE问题(Asymmetric MLWE, AMLWE) 和非对称MSIS问题(Asymmetric MSIS, AMSIS) 来设计具体的密码系统。特别地，通过在签名方案中充分利用AMSIS和AMLWE的非对称特性（详见埃奎斯签名方案文档第1.1节的设计原理部分），我们给出了比文献中已有的格上签名方案具有更好综合性能的数字签名方案，并将之命名为埃奎斯签名方案(Aigis-sig)；通过在密钥封装方案中充分利用AMLWE困难问题的非对称特性（详见埃奎斯密钥封装机制文档第1.1节的设计原理部分），我们给出了比文献中已有格上密钥封装方案具有更好综合性能的密钥封装方案，并将之命名为埃奎斯密钥封装机制(Aigis-enc)。

## 1.1 埃奎斯密钥封装机制的设计原理

为了表达得更加清楚和简洁，我们将以基于(A)LWE问题的公钥加密方案为例来阐述埃奎斯-密钥封装机制(Aigis-enc)的核心设计原理。

---

<sup>2</sup> 实际上，对于特定的分布（例如高斯分布[35]），我们也可以从理论上证明这种困难关系成立。

自Regev[35]提出LWE问题之后，世界各国密码研究学者已经基于LWE提出了许多公钥密码方案。但除了参数选择不同之外，大多基于LWE问题的公钥加密方案都基于文献[27]中的加密框架。令 $n, q \in \mathbb{Z}$ 为正整数， $\alpha \in \mathbb{R}$ 为正实数， $\chi_\alpha \subset \mathbb{Z}$ 是以 $\alpha \in \mathbb{R}$ 为标准差的高斯分布（综合考虑效率和安全性等，我们将在实际方案中使用对应参数的二项分布），基于 $LWE_{n,m,q,\alpha}$ 问题的公钥加密方案如下：

- **密钥生成算法：**随机选择矩阵 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times n}$ ，计算 $\mathbf{b} = \mathbf{As} + \mathbf{e}$ ，最终输出公钥 $pk = (\mathbf{A}, \mathbf{b})$ 和私钥 $sk = \mathbf{s}$ ，其中 $\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi_\alpha^n$ 。
- **加密算法：**给定公钥 $pk = (\mathbf{A}, \mathbf{b})$ 和明文消息 $\mu \in \{0, 1\}$ ，计算 $\mathbf{c}_1 = \mathbf{A}^T \mathbf{r} + \mathbf{x}_1$ ， $c_2 = \mathbf{b}^T \mathbf{r} + x_2 + \mu \cdot \lceil \frac{q}{2} \rceil$ ，最终输出密文 $C = (\mathbf{c}_1, c_2)$ ，其中 $\mathbf{r}, \mathbf{x}_1 \xleftarrow{\$} \chi_\alpha^n, x_2 \xleftarrow{\$} \chi_\alpha$ 。
- **解密算法：**给定私钥 $sk = \mathbf{s}$ 和密文 $C = (\mathbf{c}_1, c_2)$ ，计算 $z = c_2 - \mathbf{s}^T \mathbf{c}_1$ ，输出 $\lceil z \cdot \frac{2}{q} \rceil \bmod 2$ 。

显然，对于正常加密 $\mu \in \{0, 1\}$ 的密文 $C = (\mathbf{c}_1, c_2)$ ，我们有

$$z = c_2 - \mathbf{s}^T \mathbf{c}_1 = \mu \cdot \lceil \frac{q}{2} \rceil + \underbrace{\mathbf{e}^T \mathbf{r} - \mathbf{s}^T \mathbf{x}_1 + x_2}_{\text{解密噪音项 } e'} \quad (1)$$

成立。为了解密算法的正确性，我们必须要求噪音项 $|e'| < \frac{q}{4}$ 。由于 $|x_2|$ 的值远远小于 $|\mathbf{e}^T \mathbf{r} - \mathbf{s}^T \mathbf{x}_1|$ ， $|e'|$ 的大小主要取决于 $|\mathbf{e}^T \mathbf{r} - \mathbf{s}^T \mathbf{x}_1|$ 的大小。也就是说，LWE的秘密向量及其对应的噪音向量在最终噪音项中起着几乎同等对称的作用。进一步，当固定 $n$ 时， $|\mathbf{e}^T \mathbf{r} - \mathbf{s}^T \mathbf{x}_1|$ 值与 $\alpha$ 值正相关：

$$\begin{aligned} \alpha \text{越大} &\Rightarrow |\mathbf{e}^T \mathbf{r} - \mathbf{s}^T \mathbf{x}_1| \text{越大} \Rightarrow |e'| \text{越大;} \\ \alpha \text{越小} &\Rightarrow |\mathbf{e}^T \mathbf{r} - \mathbf{s}^T \mathbf{x}_1| \text{越小} \Rightarrow |e'| \text{越小。} \end{aligned}$$

换句话说，当固定 $n$ 时，从正确性的角度我们希望 $\alpha/q$ 越小越好。但同时，标准LWE问题的实际安全性又与 $\alpha/q$ 正相关： $\alpha/q$ 越大，相应LWE问题越困难。这种正确性和安全性在具体参数选择时的制约关系让我们选择的参数很难恰好同时满足正确性和安全性的要求。特别地，当固定参数 $n, q$ 时，我们不能在不影响安全性的前提下通过减少 $\alpha$ 值来提高方案的正确率，同时我们也不能在不影响正确性的前提下通过增大 $\alpha$ 值来提高方案的安全性。

为了提高方案的通信效率，大多数方案都会选择使用模切换技术[15,13]来压缩公钥和密文的长度。特别地，对于任意正整数 $q, p \in \mathbb{Z}$ ，定义从 $\mathbb{Z}_q$ 到 $\mathbb{Z}_p$ 的映射 $\lceil \cdot \rceil_{q \rightarrow p} : x \in \mathbb{Z}_q \rightarrow \lceil x \cdot p/q \rceil \bmod p$ 。给定公钥压缩参数 $p_1 \in \mathbb{Z}$ ，密文压缩参数 $p_2, p_3$ ，我们可以按如下方式压缩公钥和密文的长度：

- **密钥生成算法：**随机选择矩阵 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times n}$ ，计算 $\mathbf{b} = \mathbf{As} + \mathbf{e}$ ，最终输出公钥 $pk = (\mathbf{A}, \bar{\mathbf{b}} = \lceil \mathbf{b} \rceil_{q \rightarrow p_1})$ 和私钥 $sk = \mathbf{s}$ ，其中 $\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi_\alpha^n$ 。

- **加密算法:** 给定公钥  $pk = (\mathbf{A}, \bar{\mathbf{b}})$  和明文消息  $\mu \in \{0, 1\}$ , 计算  $\mathbf{c}_1 = \mathbf{A}^T \mathbf{r} + \mathbf{x}_1$ ,  $c_2 = [\bar{\mathbf{b}}]_{p_1 \rightarrow q}^T \mathbf{r} + x_2 + \mu \cdot \lceil \frac{q}{2} \rceil$ , 最终输出密文  $C = (\bar{\mathbf{c}}_1 = [\mathbf{c}_1]_{q \rightarrow p_2}, \bar{c}_2 = [c_2]_{q \rightarrow p_3})$ , 其中  $\mathbf{r}, \mathbf{x}_1 \xleftarrow{\$} \chi_{\alpha_1}^n, x_2 \xleftarrow{\$} \chi_{\alpha_2}$ 。
- **解密算法:** 给定私钥  $sk = \mathbf{s}$  和密文  $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$ , 计算  $z = [\bar{c}_2]_{p_3 \rightarrow q} - \mathbf{s}^T [\bar{\mathbf{c}}_1]_{p_2 \rightarrow q}$ , 输出  $[z]_{q \rightarrow 2} = [z \cdot \frac{2}{q}] \bmod 2$ 。

令  $\bar{\mathbf{e}} = [[\mathbf{b}]_{q \rightarrow p_1}]_{p_1 \rightarrow q} - \mathbf{b}$ ,  $\bar{\mathbf{x}}_1 = [[\mathbf{c}_1]_{q \rightarrow p_2}]_{p_2 \rightarrow q} - \mathbf{c}_1$ ,  $\bar{x}_2 = [[c_2]_{q \rightarrow p_3}]_{p_3 \rightarrow q} - c_2$ , 易证  $\|\bar{\mathbf{e}}\|_\infty \leq \frac{q}{2p_1}, \|\bar{\mathbf{x}}_1\|_\infty \leq \frac{q}{2p_2}, |\bar{x}_2| \leq \frac{q}{2p_3}$ 。对于正常加密  $\mu \in \{0, 1\}$  的密文  $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$ , 我们有

$$z = [\bar{c}_2]_{p_3 \rightarrow q} - \mathbf{s}^T [\bar{\mathbf{c}}_1]_{p_2 \rightarrow q} = \mu \cdot \lceil \frac{q}{2} \rceil + \underbrace{(\mathbf{e} + \bar{\mathbf{e}})^T \mathbf{r} - \mathbf{s}^T (\mathbf{x}_1 + \bar{\mathbf{x}}_1) + (x_2 + \bar{x}_2)}_{\text{解密噪音项 } e'} \quad (2)$$

显然, 压缩参数  $p_1, p_2, p_3$  取值越小, 公钥和密文的长度就越小。但由  $\bar{\mathbf{e}}, \bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2$  的定义和等式(2)可知当参数  $p_1, p_2, p_3$  取值越小时, 噪音项却变得越来越大。特别地, 当  $p_1, p_2, p_3$  取值变小时, 我们将会有  $\|\bar{\mathbf{e}}\|_\infty \gg \|\mathbf{e}\|_\infty, \|\bar{\mathbf{x}}_1\|_\infty \gg \|\mathbf{x}_1\|_\infty, |\bar{x}_2| \gg |x_2|$ , 从而使得  $(\mathbf{e}, \mathbf{x}_1, x_2)$  在解密噪音项中的作用将越来越小。换句话说, 由于压缩技术的使用, LWE 的秘密向量及其对应的噪音向量在最终噪音项中起着不对等的作用。特别地, 对于特定选择的压缩参数  $(p_1, p_2, p_3)$ , 减小或增大  $\|\mathbf{s}\|_\infty$  和  $\|\mathbf{r}\|_\infty$  的值将会极大地减小或增大解密噪音项  $|e'|$  的值, 但改变  $\|\mathbf{e}\|_\infty, \|\mathbf{x}_1\|_\infty$  和  $|x_2|$  的值却不会引起  $|e'|$  明显地变化。

以上非对称的现象是我们选择用非对称LWE问题来设计公钥加密方案的主要原因。正式地, 令  $\chi_{\alpha_1}$  和  $\chi_{\alpha_2}$  分别为ALWE问题中秘密向量分布和噪音向量分布。我们的公钥加密算法方案定义如下:

- **密钥生成算法:** 随机选择矩阵  $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times n}$ , 计算  $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$ , 最终输出公钥  $pk = (\mathbf{A}, \bar{\mathbf{b}} = [\mathbf{b}]_{q \rightarrow p_1})$ , 私钥  $sk = \mathbf{s}$ , 其中  $\mathbf{s} \xleftarrow{\$} \chi_{\alpha_1}^n, \mathbf{e} \xleftarrow{\$} \chi_{\alpha_2}^n$ 。
- **加密算法:** 给定公钥  $pk = (\mathbf{A}, \bar{\mathbf{b}})$  和明文消息  $\mu \in \{0, 1\}$ , 计算  $\mathbf{c}_1 = \mathbf{A}^T \mathbf{r} + \mathbf{x}_1$ ,  $c_2 = [\mathbf{b}]_{p_1 \rightarrow q}^T \mathbf{r} + x_2 + \mu \cdot \lceil \frac{q}{2} \rceil$ , 最终输出密文  $C = (\bar{\mathbf{c}}_1 = [\mathbf{c}_1]_{q \rightarrow p_2}, \bar{c}_2 = [c_2]_{q \rightarrow p_3})$ , 其中  $\mathbf{r} \xleftarrow{\$} \chi_{\alpha_1}^n, \mathbf{x}_1 \xleftarrow{\$} \chi_{\alpha_2}^n, x_2 \xleftarrow{\$} \chi_{\alpha_2}$ 。
- **解密算法:** 给定私钥  $sk = \mathbf{s}$  和密文  $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$ , 计算  $z = [\bar{c}_2]_{p_3 \rightarrow q} - \mathbf{s}^T [\bar{\mathbf{c}}_1]_{p_2 \rightarrow q}$ , 输出  $[z]_{q \rightarrow 2} = [z \cdot \frac{2}{q}] \bmod 2$ 。

同样地, 对于正常加密  $\mu \in \{0, 1\}$  的密文  $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$ , 我们有

$$z = [\bar{c}_2]_{p_3 \rightarrow q} - \mathbf{s}^T [\bar{\mathbf{c}}_1]_{p_2 \rightarrow q} = \mu \cdot \lceil \frac{q}{2} \rceil + \underbrace{(\mathbf{e} + \bar{\mathbf{e}})^T \mathbf{r} - \mathbf{s}^T (\mathbf{x}_1 + \bar{\mathbf{x}}_1) + (x_2 + \bar{x}_2)}_{\text{解密噪音项 } e'} \quad (3)$$

成立。注意到, 等式(3)中  $\|\mathbf{s}\|_\infty$  和  $\|\mathbf{r}\|_\infty$  的值只与  $\alpha_1$  的取值有关, 而  $\|\mathbf{e}\|_\infty, \|\mathbf{x}_1\|_\infty$  和  $|x_2|$  的值只与  $\alpha_2$  的值有关。直观上, 我们希望通过取较小的  $\alpha_1$  的值来得到较小解密噪音项  $|e'|$ , 但同时取较大的  $\alpha_2$  的值来弥补由于减小  $\alpha_1$  而带来的潜在安全损失。

虽然以上想法在理论上是可行的，但却对选取方案的具体参数没有太多指导意义。为此，我们考虑了目前文献中求解(A)LWE问题最有效的方法，并得到了如下实验结论：当 $\chi_{\alpha_1}$ 和 $\chi_{\alpha_2}$ 是分别以 $\alpha_1 \in \mathbb{R}$ 和 $\alpha_2 \in \mathbb{R}$ 为标准差的亚高斯分布时，ALWE问题和LWE问题困难性之间的存在如下近似关系

$$\text{ALWE}_{n,m,q,\alpha_1,\alpha_2} \approx \text{LWE}_{n,m,q,\sqrt{\alpha_1\alpha_2}}$$

显然，当 $\alpha_1 = \alpha_2$ 时，以上关系恒成立。此外，以上关系还给出了一个有用的性质：只要保持 $\alpha_1\alpha_2$ 值不变，变化 $\alpha_1, \alpha_2$ 的值并不会影响  $\text{ALWE}_{n,m,q,\alpha_1,\alpha_2}$  的困难性。换句话说，通过ALWE问题来设计更高效率的公钥加密方案是可行的。

显然，以上技术可以很自然地推广到利用RLWE和MLWE问题的所有方案。为了取得较好的计算和通信效率折衷，我们将最终用AMLWE问题来设计具体方案，并通过综合考虑所有参数的取值来实现较高的综合性能。事实上，与格上同类方案（例如[6,11]等）相比我们的方案的确能实现更好的综合效率，特别是拥有更短的公钥、私钥和密文长度。此外，即使在没有使用压缩技术的(M/R)LWE的方案中使用非对称(M/R)LWE困难问题也能够提供更灵活更细粒度的参数选择。

## 1.2 文档结构

第2节将介绍一些基本符号和操作。第3节将给出埃奎斯密钥封装机制的具体描述。第4节将给出埃奎斯密钥封装机制的实现和性能。第5节给出方案的安全证明。第6节分析了算法抗已知攻击的能力。第7节对算法的优缺点进行了分析。

## 2 符号及参数

$\mathcal{B}$	8比特无符号整数的集合(字节)，即 $\{0, \dots, 255\}$
$\mathcal{B}^k$	$\underbrace{\mathcal{B} \times \dots \times \mathcal{B}}_k$
$\mathcal{B}^*$	$\mathcal{B}^1 \cup \mathcal{B}^2 \cup \dots \cup \mathcal{B}^i \cup \dots$
$a_i$	对于字节或比特数组 $a$ ， $a_i$ 表示数组 $a$ 的第 $i$ 个元素(索引从0开始)
$a + k$	对于长度为 $\ell$ 的字节数组 $a$ 和整数 $k \in \{0, 1, \dots, \ell - 1\}$ ， $a + k$ 表示从 $a$ 的第 $k$ 个元素开始的字节数组
$\parallel$	对于字符串(或字节数组) $a$ 和 $b$ ， $a \parallel b$ 表示 $a$ 与 $b$ 的级联
$:=$	赋值操作， $a := b$ 表示将 $a$ 赋值作为 $b$
$\kappa$	安全参数

$\log_2$	底为 $e$ 的对数函数，其中 $e$ 是自然数2.71828...
$\log_2$	底为2的对数函数
$\text{negl}(\cdot)$	可忽略函数
$\mathbb{N}$	正整数集合 $\{1, 2, 3, \dots\}$
$\mathbb{R}$	实数集合
$\mathbb{Z}$	整数集合 $\{\dots, -2, -1, 0, 1, 2, \dots\}$
$\mathbb{Z}_q$	商环 $\mathbb{Z}/q\mathbb{Z}$ , 其中 $q \in \mathbb{N}$ 为正整数
$\mathbb{Z}_q^n$	$\underbrace{\mathbb{Z}_q \times \dots \times \mathbb{Z}_q}_n$
$\lceil \cdot \rceil$	向上取整函数, 对于实数 $x \in \mathbb{R}$ , $\lceil x \rceil$ 表示大于等于 $x$ 的最小整数
$\lfloor \cdot \rfloor$	向下取整函数, 对于实数 $x \in \mathbb{R}$ , $\lfloor x \rfloor$ 表示小于等于 $x$ 的最大整数
$\lceil \cdot \rceil$	向上最近取整函数, 对于实数 $x \in \mathbb{R}$ , $\lceil x \rceil$ 表示最接近 $x$ 的整数, 当上下两个整数一样近时, 该函数向上取整
$\lceil \cdot \rceil_{q \rightarrow p}$	模切换函数, 对于正整数 $q, p$ 和整数 $u \in \mathbb{Z}_q$ , $\lceil u \rceil_{q \rightarrow p} = \lceil (p/q) \cdot u \rceil \bmod p$
$\mathbf{x} \xleftarrow{\$} D$	根据分布 $D$ , 选取 $\mathbf{x}$
$\mathbf{x} \xleftarrow{\$} \mathcal{S}$	从一个集合 $\mathcal{S}$ 中均匀随机选取 $\mathbf{x}$
$\circ$	逐点相乘。对于 $\mathbf{x} = (x_1, x_2, \dots, x_n)^T \in \mathbb{Z}_q^n$ 和 $\mathbf{y} = (y_1, y_2, \dots, y_n)^T \in \mathbb{Z}_q^n$ , $\mathbf{x} \circ \mathbf{y} = (x_1 y_1, x_2 y_2, \dots, x_n y_n)^T \in \mathbb{Z}_q^n$
$R$	商环 $R = \mathbb{Z}[X]/(X^n + 1)$ , 其中 $n = 2^{n'-1}$ 使得 $X^n + 1$ 是 $2^{n'}$ 次分圆多项式
$R_q$	商环 $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ , 其中 $n = 2^{n'-1}$ 使得 $X^n + 1$ 是 $2^{n'}$ 次分圆多项式
$R^k$	$\underbrace{R \times \dots \times R}_k$
$R_q^k$	$\underbrace{R_q \times \dots \times R_q}_k$
$a, b, \dots$	环 $R$ 或 $R_q$ (包括环 $\mathbb{Z}$ 和 $\mathbb{Z}_q$ )中的元素用小写字母表示
$\mathbf{a}, \mathbf{b}, \dots$	粗体小写字母表示系数在 $R$ 或 $R_q$ 中的列向量
$\mathbf{A}, \mathbf{B}$	粗体大写字母表示矩阵
$\mathbf{a}^T$ 或 $\mathbf{A}^T$	向量 $\mathbf{a}$ 或矩阵 $\mathbf{A}$ 的转置
$\mathbf{a}[i]$	向量 $\mathbf{a}$ 的第 $i$ 个元素(索引从0开始)
$\mathbf{A}[i][j]$	矩阵 $\mathbf{A}$ 第 $i$ 行, 第 $j$ 列的元素(索引从0开始)
BytesToBits	函数BytesToBits输入长度为 $\ell$ 的字节数组, 输出长度为 $8\ell$ 的比特数组, 即 对于 $\alpha = \text{BytesToBits}(a)$ , 那么 $\alpha_i := (\lfloor a_{[i/8]} / 2^{(i \bmod 8)} \rfloor \bmod 2)$

## 2.1 基本定义

**可忽略函数.** 可忽略函数是一个函数 $\text{negl} : \mathbb{N} \rightarrow [0, 1]$ 满足对于每一个正整数 $c$ , 存在一个整数 $K$ , 满足对于所有 $k > K$ 使得 $\text{negl}(k) < 1/k^c$ 成立。

**统计距离.** 两个概率分布 $X$ 和 $Y$ 之间的统计距离定义为:

$$\Delta = \frac{1}{2} \cdot \sum_{\alpha} |\Pr[X = \alpha] - \Pr[Y = \alpha]|$$

如果 $X$ 与 $Y$ 之间的统计距离是可忽略的, 那么我们称 $X$ 统计接近于 $Y$ 。

## 2.2 基本操作

**模约化.** 对于一个正偶数 $\alpha$ , 定义 $r' = r \bmod^{\pm} \alpha$ 是在 $(-\frac{\alpha}{2}, \frac{\alpha}{2}]$ 范围内的唯一元素 $r'$ 满足 $r' = r \bmod \alpha$ 成立。对于一个正奇数 $\alpha$ , 定义 $r' = r \bmod^{\pm} \alpha$ 是在 $[-\frac{\alpha-1}{2}, \frac{\alpha-1}{2}]$ 范围内的唯一元素 $r'$ 满足 $r' = r \bmod \alpha$ 成立。对于任意正整数 $\alpha$ , 定义 $r' = r \bmod^+ \alpha$ 是在 $[0, \alpha)$ 范围内唯一的元素 $r'$ 满足 $r' = r \bmod \alpha$ 成立。当精确的表示不重要的时候, 简写为 $r \bmod \alpha$ 。

**元素的范数.** 对于一个元素 $w \in \mathbb{Z}_q$ ,  $\|w\|_\infty$ 表示 $|w \bmod^{\pm} q|$ 。下面定义关于商环 $R$ 上元素 $w = w_0 + w_1X + \dots + w_{n-1}X^{n-1} \in R$ 的 $\ell_\infty$ 和 $\ell_2$ 范数:

$$\|w\|_\infty = \max_i \|w_i\|_\infty, \quad \|w\| = \sqrt{\|w_0\|_\infty^2 + \dots + \|w_{n-1}\|_\infty^2}$$

相应地, 对于向量 $\mathbf{w} = (w_1, \dots, w_k) \in R^k$ , 定义

$$\|\mathbf{w}\|_\infty = \max_i \|w_i\|_\infty, \quad \|\mathbf{w}\| = \sqrt{\|w_1\|^2 + \dots + \|w_k\|^2}$$

**模切换函数.** 对于任意正整数 $p, q$ , 定义模切换函数 $\lceil \cdot \rceil_{q \rightarrow p}$ 如下:

$$\lceil x \rceil_{q \rightarrow p} = \lceil (p/q) \cdot x \rceil \bmod^+ p$$

易证, 对于任意 $x \in \mathbb{Z}_q$ ,  $p < q$ 和 $x' = \lceil \lceil x \rceil_{q \rightarrow p} \rceil_{p \rightarrow q}$ , 我们有 $|x' - x \bmod^{\pm} q| \leq B_q := \left\lceil \frac{q}{2p} \right\rceil$ 成立。当我们操作 $\lceil \cdot \rceil_{q \rightarrow p}$ 作用于环元素 $x \in R_q$ 或 $\mathbf{x} \in R_q^k$ 的时, 其意思是将相应计算过程独立作用到环元素的每个系数上。

**对称密码组件.** 埃奎斯密钥封装机制需要用到一个伪随机函数 $\text{PRF} : \mathcal{B}^\ell \times \mathcal{B} \rightarrow \mathcal{B}^*$ , 一个扩展输出函数 $\text{XOF} : \mathcal{B}^* \rightarrow \mathcal{B}^*$ , 和两个杂凑函数 $\text{H} : \mathcal{B}^* \rightarrow \mathcal{B}^\ell$ 和 $\text{G} : \mathcal{B}^* \rightarrow \mathcal{B}^\ell \times \mathcal{B}^\ell$ 。本文档仅考虑 $\ell = 32$ 或 $\ell = 64$ 。

**数论变换(NTT).** 对于商环  $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ , 我们将选择模数  $q$  使得  $\mathbb{Z}_q$  中存在一个  $2n$  次单位根  $r \bmod q$ 。在这种情况下, 分圆多项式  $X^n + 1$  完全分解为线性因子  $X - (r^i \bmod q)$  对于  $i = 1, 3, 5, \dots, 2n - 1$ 。根据中国剩余定理(CRT), 分圆环  $R_q$  与环  $\mathbb{Z}_q[X]/(X - r^i) \cong \mathbb{Z}_q$  的直积同构, 即  $R_q \cong \prod_i \mathbb{Z}_q[X]/(X - r^i)$ 。此外, 我们能够利用快速傅里叶变换(FFT)快速计算该同构

$$a \mapsto (a(r), a(r^3), \dots, a(r^{2n-1})) : R_q \rightarrow \prod_i \mathbb{Z}_q[X]/(X - r^i)$$

当基域是有限域的时候, FFT 也称为 NTT。我们实现的快速 NTT 算法并没有按照顺序  $a(r), a(r^3), \dots, a(r^{2n-1})$  输出, 而是输出

$$\hat{a} = \text{NTT}(a) = (a(r_0), a(-r_0), \dots, a(r_{n/2-1}), a(-r_{n/2-1}))$$

其中  $r_i = r^{\text{brv}(n/2+i)}$  和  $\text{brv}(k)$  表示  $k(\log_2 n)$  比特的比特逆序。利用 NTT 变换及其逆变换  $\text{NTT}^{-1}$ , 我们可以快速的计算  $R_q$  中的元素乘法。特别地, 对于  $a, b \in R_q$ , 我们有  $a \cdot b = \text{NTT}^{-1}(\text{NTT}(a) \circ \text{NTT}(b))$ 。对于向量  $\mathbf{v}$  和矩阵  $\mathbf{A}$ ,  $\hat{\mathbf{v}} = \text{NTT}(\mathbf{v})$  和  $\hat{\mathbf{A}} = \text{NTT}(\mathbf{A})$  代表将 NTT 变换独立的作用到多项式向量或矩阵中的每一个多项式。

**均匀随机采样  $R_q$  中的多项式.** 我们将采用算法 1 中的函数  $\text{Parse} : \mathcal{B}^* \rightarrow R_q$  来确定性地采样  $R_q$  中的元素, 其中  $\lceil \log_2 q \rceil \leq 16$ 。特别地。该函数以一个字节流  $B = b_0, b_1, b_2, \dots$  作为输入, 然后输出  $R_q$  中的一个元素  $\hat{a} = \hat{a}_0 + \hat{a}_1 X + \dots + \hat{a}_{n-1} X^{n-1}$ 。由于 NTT 变换能将  $R_q$  中均匀随机分布的元素映射到  $\mathbb{Z}_q^n$  中均匀随机分布的向量, 我们可以假设函数  $\text{Parse}$  的输出即为某个随机元素经过 NTT 变换后的结果。

---

**算法 1: Parse :  $\mathcal{B}^* \rightarrow R_q$**

---

```

输入: 字节流  $B = b_0, b_1, b_2, \dots \in \mathcal{B}^*$ 
输出: 多项式  $\hat{a} \in R_q$ 

1  $i := 0;$ 
2  $j := 0;$ 
3 while  $j < n$  do
4    $d := b_i + 256 \cdot b_{i+1};$ 
5    $d := d \bmod^+ 2^{\lceil \log_2 q \rceil};$ 
6   if  $d < q$  then
7      $\hat{a}_j := d;$ 
8      $j := j + 1;$ 
9   end
10   $i := i + 2;$ 
11 end
12 return  $\hat{a}_0 + \hat{a}_1 X + \dots + \hat{a}_{n-1} X^{n-1};$ 

```

---

**二项分布采样.** 以正整数 $\eta$ 为参数的中心二项分布 $B_\eta$ 定义如下:

$$B_\eta = \left\{ \sum_{i=1}^{\eta} (a_i - b_i) : (a_1, \dots, a_\eta, b_1, \dots, b_\eta) \xleftarrow{\$} \{0, 1\}^{2\eta} \right\}$$

从 $B_\eta$ 中采样一个多项式 $f \in R_q$ 或多项式向量的意思是从 $B_\eta$ 中采样每个多项式的系数。特别地，我们将利用算法2中的函数 $CBD_\eta$ (Centered Binomial Distribution)来采样满足分布 $B_\eta$ 的多项式 $f \in R_q$ ，其中 $\eta \leq 8$ 。

---

**算法 2:**  $CBD_\eta : \mathcal{B}^{n\eta/4} \rightarrow R_q$

---

输入: 字节数组 $B = (b_0, b_1, \dots, b_{n\eta/4-1}) \in \mathcal{B}^{n\eta/4}$

输出: 多项式 $f \in R_q$

```

1  $(\beta_0, \dots, \beta_{2n\eta-1}) := \text{BytesToBits}(B);$ 
2 for  $i$ 从0到 $n-1$  do
3    $a := \sum_{j=0}^{\eta-1} \beta_{2i\eta+j};$ 
4    $b := \sum_{j=0}^{\eta-1} \beta_{2i\eta+\eta+j};$ 
5    $f_i := a - b;$ 
6 end
7 return  $f_0 + f_1X + \dots + f_{n-1}X^{n-1};$ 
```

---

埃奎斯密钥封装机制还需要从 $\eta = 12$ 的中心二项分布 $B_{12}$ 中采样多项式。为了便于实现，我们将利用关系 $B_{12} = B_8 + B_4$ 来定义函数 $CBD_{12}(\mathcal{B}^{3n}) \rightarrow R_q$ :

$$CBD_{12}(B \| B' \in \mathcal{B}^{3n}) = CBD_8(B \in \mathcal{B}^{2n}) + CBD_4(B' \in \mathcal{B}^n)$$

**编码与解码.** 算法3定义了一个将 $n\ell/8$ 字节数组解码为多项式 $f = f_0 + f_1X + \dots + f_{n-1}X^{n-1}$ 的解码函数，其中每个系数 $f_i \in \{0, 1, \dots, 2^\ell - 1\}$ 。此外，我们定义编码函数 $Encode_\ell$ 为 $Decode_\ell$ 的逆。当 $Encode_\ell$ 输入为一个多项式向量时，我们独立应用 $Encode_\ell$ 到每个分量的多项式，然后将字节数组级联输出（对应的 $Decode_\ell$ 函数将按顺序逐一恢复向量中的每个多项式分量）。

---

**算法 3:**  $Decode_\ell : \mathcal{B}^{n\ell/8} \rightarrow R_q$

---

输入: 字节数组 $B \in \mathcal{B}^{n\ell/8}$

输出: 多项式 $f \in R_q$

```

1  $(\beta_0, \dots, \beta_{n\ell-1}) := \text{BytesToBits}(B);$ 
2 for  $i$ 从0到 $n-1$  do
3    $f_i := \sum_{j=0}^{\ell-1} \beta_{i\ell+j} 2^j;$ 
4 end
5 return  $f_0 + f_1X + \dots + f_{n-1}X^{n-1};$ 
```

---

### 3 埃奎斯密钥封装机制

埃奎斯密钥封装机制(Aigis-enc)本质上是基于文献[11,27]中密钥封装机制,其主要区别在于底层使用了不同的困难问题和不同的参数选取方式。为了便于描述,我们将先给出选择明文攻击安全的埃奎斯公钥加密方案作为中间方案。

#### 3.1 选择明文攻击安全的公钥加密方案

埃奎斯公钥加密方案需要由8个参数 $n, q, k, \eta_1, \eta_2, d_t, d_u, d_v$ 来实例化。第3.3节将给出方案的具体参数选择。我们将在算法4中给出埃奎斯公钥加密方案的密钥生成算法Aigis-pke.KeyGen(),在算法5中给出加密算法Aigis-pke.Enc(),在算法6中给出解密算法Aigis-pke.Dec()。

---

##### 算法 4: Aigis-pke.KeyGen(): 密钥生成算法

---

```

输出: 公钥  $pk \in \mathcal{B}^{d_t \cdot k \cdot n / 8 + n / 8}$ 
输出: 私钥  $sk \in \mathcal{B}^{\lceil \log_2 q \rceil \cdot k \cdot n / 8}$ 

1  $d \xleftarrow{\$} \mathcal{B}^{n/8};$ 
2  $(\rho, \sigma) := \mathsf{G}(d) \in \mathcal{B}^{n/4};$ 
3  $N := 0;$ 
4 for  $i$ 从0到 $k - 1$  do  $/*$  生成矩阵  $\hat{\mathbf{A}} = \mathsf{NTT}(\mathbf{A}) \in R_q^{k \times k}$   $*/$ 
5   for  $j$ 从0到 $k - 1$  do
6      $\hat{\mathbf{A}}[i][j] := \mathsf{Parse}(\mathsf{XOF}(\rho \| j \| i))$ 
7   end
8 end
9 for  $i$ 从0到 $k - 1$  do  $/*$  从  $B_{\eta_1}$  中采样  $\mathbf{s} \in R_q^k$   $*/$ 
10    $\mathbf{s}[i] := \mathsf{CBD}_{\eta_1}(\mathsf{PRF}(\sigma, N));$ 
11    $N := N + 1;$ 
12 end
13 for  $i$ 从0到 $k - 1$  do  $/*$  从  $B_{\eta_2}$  中采样  $\mathbf{e} \in R_q^k$   $*/$ 
14    $\mathbf{e}[i] := \mathsf{CBD}_{\eta_2}(\mathsf{PRF}(\sigma, N));$ 
15    $N := N + 1;$ 
16 end
17  $\hat{\mathbf{s}} := \mathsf{NTT}(\mathbf{s});$ 
18  $\mathbf{t} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \circ \hat{\mathbf{s}}) + \mathbf{e};$   $/*$   $\mathbf{t} := \mathbf{As} + \mathbf{e}$   $*/$ 
19  $pk := (\mathsf{Encode}_{d_t}(\lceil \mathbf{t} \rceil_{q \rightarrow 2^{d_t}}) \| \rho);$ 
20  $sk := \mathsf{Encode}_{\lceil \log_2 q \rceil}(\hat{\mathbf{s}} \bmod^+ q);$   $/*$   $sk := \mathbf{s}$   $*/$ 
21 return  $(pk, sk);$ 

```

---

---

**算法 5: Aegis-pke.Enc( $pk, \mu; r$ ): 加密算法**


---

输入: 公钥  $pk \in \mathcal{B}^{d_t \cdot k \cdot n/8 + n/8}$ , 明文消息  $\mu \in \mathcal{B}^{n/8}$ , 随机数  $r \in \mathcal{B}^{n/8}$

输出: 密文  $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$

```

1  $N := 0;$ 
2  $\mathbf{t} := \lceil \text{Decode}_{d_t}(pk) \rceil_{2^{d_t} \rightarrow q};$ 
3  $\rho := pk + d_t \cdot k \cdot n/8;$ 
4 for  $i$ 从0到 $k - 1$  do /* 生成矩阵  $\hat{\mathbf{A}} = \text{NTT}(\mathbf{A}) \in R_q^{k \times k}$  */
5   for  $j$ 从0到 $k - 1$  do
6      $\hat{\mathbf{A}}[i][j] := \text{Parse}(\text{XOF}(\rho \| j \| i))$ 
7   end
8 end
9 for  $i$ 从0到 $k - 1$  do /* 从  $B_{\eta_1}$  中采样  $\mathbf{r} \in R_q^k$  */
10    $\mathbf{r}[i] := \text{CBD}_{\eta_1}(\text{PRF}(r, N));$ 
11    $N := N + 1;$ 
12 end
13 for  $i$ 从0到 $k - 1$  do /* 从  $B_{\eta_2}$  中采样  $\mathbf{e}_1 \in R_q^k$  */
14    $\mathbf{e}_1[i] := \text{CBD}_{\eta_2}(\text{PRF}(r, N));$ 
15    $N := N + 1;$ 
16 end
17  $e_2 := \text{CBD}_{\eta_2}(\text{PRF}(r, N));$  /* 从  $B_{\eta_2}$  中采样  $e_2 \in R_q$  */
18  $\hat{\mathbf{r}} := \text{NTT}(\mathbf{r});$ 
19  $\mathbf{u} := \text{NTT}^{-1}(\hat{\mathbf{A}}^T \circ \hat{\mathbf{r}}) + \mathbf{e}_1;$  /*  $\mathbf{u} := \mathbf{A}^T \mathbf{r} + \mathbf{e}_1$  */
20  $v := \text{NTT}^{-1}(\text{NTT}(\mathbf{t})^T \circ \hat{\mathbf{r}}) + e_2 + \lceil \frac{q}{2} \rceil \cdot \text{Decode}_1(\mu);$ 
/*  $v := \mathbf{t}^T \mathbf{r} + e_2 + \lceil \frac{q}{2} \rceil \cdot \mu$  */
21  $c_1 := \text{Encode}_{d_u}(\lceil \mathbf{u} \rceil_{q \rightarrow 2^{d_u}});$ 
22  $c_2 := \text{Encode}_{d_v}(\lceil v \rceil_{q \rightarrow 2^{d_v}});$ 
23 return  $c = (c_1 \| c_2);$ 

```

---

---

**算法 6: Aegis-pke.Dec( $sk, c$ ): 解密算法**


---

输入: 私钥  $sk \in \mathcal{B}^{\lceil \log_2 q \rceil \cdot k \cdot n / 8}$ , 密文  $c \in \mathcal{B}^{d_u \cdot k \cdot n / 8 + d_v \cdot n / 8}$

输出: 明文消息  $\mu \in \mathcal{B}^{n/8}$

- 1  $\mathbf{u} := \lceil \text{Decode}_{d_u}(c) \rceil_{2^{d_u} \rightarrow q};$
  - 2  $v := \lceil \text{Decode}_{d_v}(c + d_u \cdot k \cdot n / 8) \rceil_{2^{d_v} \rightarrow q};$
  - 3  $\hat{\mathbf{s}} := \text{Decode}_{\lceil \log_2 q \rceil}(sk);$
  - 4  $\mu := \text{Encode}_1(\lceil v - \text{NTT}^{-1}(\hat{\mathbf{s}}^T \circ \text{NTT}(\mathbf{u})) \rceil_{q \rightarrow 2}); \quad /* \mu := \lceil v - \mathbf{s}^T \mathbf{u} \rceil_{q \rightarrow 2} */$
  - 5 **return**  $\mu;$
- 

**正确性分析.** 令变量  $\mathbf{c}_t \in R^k$  与算法 5 中计算的向量  $\mathbf{t}$ (第2步)满足如下关系:

$$\mathbf{t} := \lceil \lceil \mathbf{A}\mathbf{s} + \mathbf{e} \rceil_{q \rightarrow 2^{d_t}} \rceil_{2^{d_t} \rightarrow q} = \mathbf{A}\mathbf{s} + \mathbf{e} - \mathbf{c}_t.$$

令变量  $\mathbf{c}_u \in R^k$  与算法 6 中计算的向量  $\mathbf{u}$ (第1步)满足如下关系:

$$\mathbf{u} := \lceil \lceil \mathbf{A}^T \mathbf{r} + \mathbf{e}_1 \rceil_{q \rightarrow 2^{d_u}} \rceil_{2^{d_u} \rightarrow q} = \mathbf{A}^T \mathbf{r} + \mathbf{e}_1 - \mathbf{c}_u.$$

令变量  $\mathbf{c}_v \in R$  与算法 6 中计算的值  $v$ (第2步)满足如下关系:

$$\begin{aligned} v &:= \lceil \lceil \mathbf{t}^T \mathbf{r} + e_2 + \lceil q/2 \rceil \cdot \mu \rceil_{q \rightarrow 2^{d_v}} \rceil_{2^{d_v} \rightarrow q} \\ &= \mathbf{t}^T \mathbf{r} + e_2 + \lceil q/2 \rceil \cdot \mu - c_v \\ &= (\mathbf{A}\mathbf{s} + \mathbf{e} - \mathbf{c}_t)^T \mathbf{r} + e_2 + \lceil q/2 \rceil \cdot \mu - c_v \\ &= (\mathbf{A}\mathbf{s} + \mathbf{e})^T \mathbf{r} + e_2 + \lceil q/2 \rceil \cdot \mu - c_v - \mathbf{c}_t^T \mathbf{r}. \end{aligned}$$

利用以上关系式, 我们有

$$v - \mathbf{s}^T \mathbf{u} = \underbrace{\mathbf{e}^T \mathbf{r} + e_2 - c_v - \mathbf{c}_t^T \mathbf{r} - \mathbf{s}^T \mathbf{e}_1 + \mathbf{s}^T \mathbf{c}_u}_{= w} + \lceil q/2 \rceil \cdot \mu = w + \lceil q/2 \rceil \cdot \mu.$$

定义  $\mu' := \lceil v - \mathbf{s}^T \mathbf{u} \rceil_{q \rightarrow 2}$ 。那么, 我有以下关系成立:

$$\lceil q/4 \rceil \geq \|v - \mathbf{s}^T \mathbf{u} - \lceil q/2 \rceil \cdot \mu'\|_\infty = \|w + \lceil q/2 \rceil \cdot \mu - \lceil q/2 \rceil \cdot \mu'\|_\infty$$

由三角不等式可知  $\|\lceil q/2 \rceil \cdot \mu - \lceil q/2 \rceil \cdot \mu'\|_\infty \leq \lceil q/4 \rceil + \|w\|_\infty$ 。换句话说, 如果  $\|w\|_\infty < \lceil q/4 \rceil$ , 那么我们有

$$\|\lceil q/2 \rceil \cdot \mu - \lceil q/2 \rceil \cdot \mu'\|_\infty < 2 \cdot \lceil q/4 \rceil.$$

容易验证, 对于所有奇数  $q$ , 上述不等式只有在  $m = m'$  时才能成立。换句话说, 只要  $\|w\|_\infty < \lceil q/4 \rceil$  成立, 那么埃奎斯公钥加密方案就能正确解密。我们将选择合适的参数(在第3.3节描述)使得解密出错的概率是可忽略的。

### 3.2 选择密文攻击安全的密钥封装机制

基于选择明文攻击安全的公钥加密方案Aigis-pke，我们将在这一节中给出选择密文攻击安全的埃奎斯密钥封装机制。算法7, 8和9分别定义了Aigis-enc 的密钥生成算法、密钥封装算法和解封装算法。

---

#### 算法 7: Aigis-enc.KeyGen(): 密钥生成算法

---

输出: 公钥  $pk \in \mathcal{B}^{d_t \cdot k \cdot n/8 + n/8}$   
 输出: 私钥  $sk \in \mathcal{B}^{(\lceil \log_2 q \rceil + d_t) \cdot k \cdot n/8 + 3n/8}$

- 1  $z \xleftarrow{\$} \mathcal{B}^{n/8};$
- 2  $(pk, sk') := \text{Aigis-pke.KeyGen}();$
- 3  $sk := (sk' \| pk \| \mathsf{H}(pk) \| z);$
- 4 **return**  $(pk, sk);$

---



---

#### 算法 8: Aigis-enc.Encaps( $pk$ ): 密钥封装算法

---

输入: 公钥  $pk \in \mathcal{B}^{d_t \cdot k \cdot n/8 + n/8}$   
 输出: 密文  $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$   
 输出: 密钥  $K \in \mathcal{B}^{n/8}$

- 1  $\mu \xleftarrow{\$} \mathcal{B}^{n/8};$
- 2  $\mu' := \mathsf{H}(\mu);$
- 3  $(\bar{K}, r) := \mathsf{G}(\mu' \| \mathsf{H}(pk));$
- 4  $c := \text{Aigis-pke.Enc}(pk, \mu'; r);$
- 5  $K := \mathsf{H}(\bar{K} \| \mathsf{H}(c));$
- 6 **return**  $(c, K);$

---

---

**算法 9: Aegis-enc.Decaps( $sk, c$ ): 解封装算法**


---

输入: 密文  $c \in \mathcal{B}^{d_u \cdot k \cdot n/8 + d_v \cdot n/8}$   
输入: 私钥  $sk \in \mathcal{B}^{(\lceil \log_2 q \rceil + d_t) \cdot k \cdot n/8 + 3n/8}$   
输出: 密钥  $K \in \mathcal{B}^{n/8}$

```

1  $pk := sk + \lceil \log_2 q \rceil \cdot k \cdot n/8;$ 
2  $h := sk + (\lceil \log_2 q \rceil + d_t) \cdot k \cdot n/8 + n/8; \quad /* h = H(pk) */$ 
3  $z := sk + (\lceil \log_2 q \rceil + d_t) \cdot k \cdot n/8 + n/4;$ 
4  $\mu' := \text{Aegis-pke.Dec}(sk, c);$ 
5  $(\bar{K}', r') := G(\mu' \| h);$ 
6  $c' := \text{Aegis-pke.Enc}(pk, \mu'; r');$ 
7 if  $c = c'$  then
8     return  $K := H(\bar{K}' \| H(c))$ 
9 else
10    return  $K := H(z \| H(c))$ 
11 end
12 return  $K;$ 

```

---

### 3.3 参数集以及相关函数的实例化

根据当前格上困难问题求解状态和未来数年内对抗量子安全密钥封装机制的需求, 我们为埃奎斯密钥封装机制选择了三组参数集, 即PARAMS I、PARAMS II 和PARAMS III, 分别瞄准目标量子安全强度80, 128和192 (对应保守估计的经典安全强度分别约为112, 162和235), 其中PARAMS II为推荐参数。由于埃奎斯密钥封装机制存在一定的解密错误, 根据目前格上求解算法对于解密错误的利用状态和量子可证明安全的要求, 我们特意选择对应参数使得方案的解密错误率 (分别达到了 $2^{-82}$ ,  $2^{-128}$ 和 $2^{-211}$ ) 与安全性相匹配, 从而保证不会由于解密错误的存在而降低方案抵抗量子敌手的安全性。此外, 由于量子搜索算法的存在,  $2\kappa$ 比特长的密钥最多只能提供 $\kappa$ 比特的安全强度, 为了实现大于128的量子安全强度, 参数集PARAMS III支持产生64字节 (即512比特) 的会话密钥, 而对应参数集PARAMS I和参数集PARAMS II只支持产生32字节 (即256比特) 的会话密钥。

**实例化PRF, XOF, H和G.** 我们可以采用FIPS-202标准[32]中的函数来实例化这些对称密码组件:

- 用SHAKE-256( $s \| m$ )实例化PRF( $s, m$ )
- 用SHAKE-128实例化XOF

表 1. 埃奎斯密钥封装机制的参数集(公钥pk、私钥sk、密文c和会话密钥ss长度的单位为字节)

参数集名称	$(n, k, q)$	$(\eta_1, \eta_2)$	$(d_t, d_u, d_v)$	公钥 pk	私钥 sk	密文 c	会话密钥 ss	解密错误率
<b>PARAMS I</b>	(256, 2, 7681)	(2, 12)	(10, 9, 3)	672	1568	672	32	$2^{-82}$
<b>PARAMS II</b>	(256, 3, 7681)	(1, 4)	(9, 9, 4)	896	2208	992	32	$2^{-128}$
<b>PARAMS III</b>	(512, 2, 12289)	(2, 8)	(11, 10, 4)	1472	3392	1536	64	$2^{-211}$

- 当 $n = 256$ 时，用SHA3-256实例化H；当 $n = 512$ 时，用SHA3-512实例化H
- 当 $n = 256$ 时，用SHA3-512实例化G；当 $n = 512$ 时，用SHA3-1024实例化G

## 4 程序实现及性能

我们分别在Windows 10 64位系统（硬件配置为2.5GHz的Intel Core-i7 6500U CPU和8GB内存的Thinkpad X1笔记本）和Ubuntu 14.04 LTS 64位操作系统（硬件配置为3.6GHz的Intel Core-i7 4790 CPU和4GB的内存ThinkCentre台式机）上实现了埃奎斯密钥封装机制，并将相关实现代码分别保存在Aegis-enc文件夹下名为“Win10”和“Ubuntu”的文件夹内。特别地，针对两种操作系统我们都分别给出了标准C语言的实现版本和利用AVX2部分优化实现的版本，并将相关实现代码分别保存在名为“ref”和“avx2”的文件内。表2和表3分别给出了埃奎斯密钥封装机制各算法在Win10和Ubuntu系统上运行10000次的平均CPU周期。

表 2. 埃奎斯密钥封装机制Win10版本的计算效率（单位：CPU周期）

参数集名称	密钥生成	密钥封装	解封装	密钥生成(AVX2)	密钥封装(AVX2)	解封装(AVX2)
<b>PARAMS I</b>	137 751	196 576	213 957	54 838	73 018	65 752
<b>PARAMS II</b>	205 866	268 938	291 920	76 778	102 015	92 686
<b>PARAMS III</b>	308 050	440 070	460 837	128 651	185 796	153 445

表 3. 埃奎斯密钥封装机制Ubuntu版本的计算效率（单位：CPU周期）

参数集名称	密钥生成	密钥封装	解封装	密钥生成(AVX2)	密钥封装(AVX2)	解封装(AVX2)
<b>PARAMS I</b>	159 764	216 617	244 179	71 352	76 583	64 634
<b>PARAMS II</b>	232 102	295 222	331 524	90 335	99 780	85 622
<b>PARAMS III</b>	335 478	460 563	506 319	146 304	174 560	140 894

### 4.1 编译和运行程序

**Windows下编译和运行程序：**用VS 2013或更新的版本打开VS项目文件即可编译并运行程序。默认情况下，编译程序将会生成配置为PARAMS II中参数的

密钥封装机制的软件。如果想要编译程序生成配置为PARAMS I或者PARAMS III中参数的密钥封装机制的软件，只需要打开名为params.h的头文件，并对应修改PARAMS标志为1或者3即可。

**Ubuntu下编译和运行程序：**用命令行终端进入程序源代码所在文件夹(例如, Ubuntu/ref)，然后运行如下命令即可编译并运行程序：

- 
- 1 make
  - 2 make test

与Windows下的程序一样，默认情况下编译程序将会生成配置为PARAMS II中参数的密码封装机制的软件。如果想要编译程序生成配置为PARAMS I或者PARAMS III中参数的密钥封装机制的软件，只需要打开名为params.h的头文件，并对应修改PARAMS标志为1或者3即可。

## 4.2 测试用例

在Ubuntu编译环境下，我们还提供测试用例生成程序。由于该程序需要用到openssl库中的某些函数，测试用例生成程序只有在系统安装了openssl库后才能被正确地生成。特别地，如果目标系统已经事先将openssl库安装到了路径 /usr/local/include 和 /usr/local/lib 下，那么执行make命令时编译程序会自动生成测试用例程序，并且可以用make kat命令运行该程序。

- 
- 1 make
  - 2 make kat

运行该命令后，在程序的文件夹下将会产生两个分别以.req和.rsp结尾的文件，其中.req文件包含一组包括测试用例编号count、随机种子seed、公钥pk、私钥sk、密文ct和共享密钥ss的数据集合（除了变量count和seed有数据之外，其他变量都为空）。以下是.req文件中的一组样本数据：

```
count = 0

seed = 061550234D158C5EC95595FE04EF7A25767F2E24CC2BC47
9D09D86DC9ABCFDE7056A8C266F9EF97ED08541DBD2E1FFA1

pk =

sk =
```

ct =

ss =

对应地，.rsp文件也包含一组由测试用例编号count、随机种子seed、公钥pk、私钥sk、密文ct和共享密钥ss的数据集合，其中.rsp中seed变量与.req中具有相同测试用例编号中的seed的值相同，而公钥pk、私钥sk、密文ct和共享密钥ss的值则是在以seed初始化一个伪随机生成器来为其他算法提供随机数的情况下由相应算法来确定性的产生。以下是.rsp文件中的一组样本数据：

count = 0

seed = 061550234D158C5EC95595FE04EF7A25767F2E24CC2BC47  
9D09D86DC9ABCFDE7056A8C266F9EF97ED08541DBD2E1FFA1

pk =BA13464ECE8CC142AB7219BD48A806B9B731851D0463C2E9  
0EC2242532330E0C68F82F751C82F5CD98E80DA0AFA63C9537803  
C3C95CD34A9B607405EE55A38590387521DF8CCDCEF33EFFE340  
9B6B83B78EE681C7335BFC7F6FFF96F48433195C09BF06E5F72EC  
DD625C6E1BDACC608A9EDB0384EDE57C65B2A533EB9E0299DB  
CFABC93E0DA39C1D9EF22C81823FF2E310F0505B33E87AC8747A  
269CEB56F49DEF3BD92B87F2B72F3A33F99683078926EA386A74E  
B1B40EF4CC4BCFA3B13544F3ED7D723727634CF8D2EB0593AE9  
EC812479F20556B321856AA963D55E57DA3680E259832A50E63DD9  
7F537CE8951B8A84A2B9D5BD4AAA8BDEEA283CAD6BBCE12B3  
7C7FC6BA7F8B035710FBAFBF5F3274B6F23D4108F8AA023A0494  
DA716F8F6829C02C1A1262351219DD2B936781090C7F5395892E12B  
2754DF6AB8E86DDB152A0FA6987FD2A49C691DFCF448026B3696  
84B9074D8222FDB8F3F8AEBED34F87E8D03278796FCBB44A8C57  
82EDE5DD5CA797009C9EF338D4EC5F5205458110F69C787C4608C  
2DF0017D3ED2A8497D861D02FADBF7B0BA225ADEE2EF3D4368  
F3886FE3EA39258A3A9F0F5CB4AFB9BF744045F4DFD80AF21877  
357EACDC2EA9CD0144603A0955A907712BA6F0A2E89257EF0E54  
3488197171CAA3B6D00E5DAE0F82256E9F1EA9B9316D2E6CBC3F  
4B4B4273F9AA108BC5216BE289289BA6C6FCC60F8EFDAABF4D  
EED6F0C126811C9380F7CC29B974E371BC492323A8AD74119E8DB

DD0D5C751C159A8CE5969EC02CF4AFE521B04E774F2F0DC7A73  
AE969F2BE0BAE7BCAC4B6787420055E34051E0DCE87AC965BAF  
2586E0CB3FDE065E3E0F6088F7A4569D85EFBCFBAE7ACD58ED  
F30C456EF81807757A939AEA0F0C6885995E72291A4BA8633A18D  
CAE58060919D5B4A88D81FDEDDF0D05825FC3428F22F4EF29FA5  
3A92054F7D3198157F13FFA9A185A5ADF74392B4E97816C843ACC  
E4F06A3DE99ADDEBEF7BC5B12E89FAF39378F0326CED96708FC  
3F052BD315B0D208C6BE48246E7090BECE67D38F29DA26BEE30A  
4E33539B25E02AAC6DB84AD6CAD0C3F89E504D0F04848AC4A14  
039DE1F8EAD2789093F3CA05A5CA799F36DF783C437163D62E6C  
8D0BDA2DD2E2DF0EC8A77898A9789B921DABA4692C5C9D1DD0  
6467468E4118620294C9D17591354B204898695D87C64A3140E0DB1D  
A31B0D376FEB982FE365EAFD465FC64A0C5F8F3F9003489415899  
D59A543D8208C54A3166529B53922

sk = 53CB1C4F5F7B7D4D67CFF760ED7341F2900538D904800F578  
29B474B0F6B008AD8A21D5930902544BAC159ECB4608BD9198D79  
7FAAAE2AB1EED2A7DC84F6B6839FBF99D32C511781D123039A2  
9D7FCC522A70EE2ECD7C3196E375A34C7AD91CF3A07EEA812F9  
C03A53C164ED6139EA1FE863EC62D96375106385266D538FCCA80  
06FA8E87C30A789086CC492C0C04E03332AEEF2DD4E9341A12C4  
09587930FFD5E5C133154D811A0BCDDD1788EBD578D65552A69E  
303FBC81B1557FA221D82CDA669A556BB9562BB6217BF5780FD5  
34181D586A94686C91146EDE8E8D44A952EE4D04281479E9A5E7D  
C2D6CD1FF04D69C21299DF46FCEF45A0EFC8BA7EE29D4030E77  
CCDE5310A6952E0BB098A33233993D8E91CE009FA7EC5C7AF0B  
72CBC6A66E1F5C6D1E9B2CCBF42BA4F571E982686A7EA126E4A  
BB66F0BDD8590E6FC3A3EF4A5E16A6843CBC509B7DCBD6B174  
B3F90479F530C4CA989C58183A11A916352BB661F85234BF550101C  
7FB0F072B4ACDA4EE7A9295752413E5CBA7F6845FFFD39827B8F  
A02387B6283BA22598337B6293BD4497988929C72D6B4E2AC24D1D  
EE25189E7E558F3C9227B39B7A2B9912AC9377F75875C4A33C68E6  
963712DBACE9D140B40D4D7DA7BDCA836FAF83619AD4945B9F8  
E9F860D1984E98396CF6E2C6FE5B7E533EEC762E42EF002DBBAD  
C0835442C4759B1F8CA327520C07DD2B04BCF5754200BB991EE96

2BB8A8EFD7FD735CD08B70560F07682B892B6C058D8B9248E808F  
B31107960C3BE0A130B4DA870281BF32804C8DD7A6BE900E93433  
E8B41636935180BBAE43764356AEECF26643FEC38CC34852D54575  
335CBA6A48C5DCD487890337C174DB62B425FE58A09F195F70369  
56B45382B2C59E294D2DBF6536CC312FCA6FBCA84F0C1E1D1CE  
3939B14E1388415D98461C9858A0780303E7169853278083CED16AA  
B3171B78DED6342BEEF6B9D76A4EAE8A0868A9ACE100682D7FD  
AD6532A6645E303F02CE0EA7B8FE3C70E5C3CE297D3AA29AEC2  
D3F1C5E0C5D6DE3B82565F56D794645D5C6C97A8C9B6BCDD4A5  
0540A7DA1511B6231354C438CDAF8CC62986162965F0656528CA44  
F10FAD7D92BC9B5AD64F2892490CD083CC4910D1E7932B82614D  
156F51085DB0B560FF828E1A819FF733E2333EF0E522688AA685B3  
EBF6B5B073D78CDB0B5370950719A4B6F5F04ECDA487CB6E135E  
D556BE7E1E1F2FC09873A6EAE2A00C699E8C5817D4337BAD1EC  
78F7E4E7D28389F195271A602E68E54594FA4CCFD0196D166D290B  
82C13C9B15A57D33C04770A620444481F736BD4030CD15659BD6B  
77D6EDF8F8041213A0E5F7848574A52747C94DC7E3D36F1A20A57  
F676C8F2046E731F0EC41BB0F314CB0175B85EEB9FE921C592802  
E660A0C91709AC85F4BCE30D35B8D56701D679642CF4302FF6136  
5A60D93B7E0692D589EBCF4623C38F315A6B6BD2C8BA6AF19360  
533F0824B89527C896624D97FCC0CED5E12C213C5ADADB5E9A45  
67B315FC84E163437151193DF7E45D52E0B2B1287E923600B3A0348  
3C59B42873CCBC40C1176A9A872159F2169B4031673255D5AAA070  
B62DEE5D009D2A12BEFE2294E4055623714D27F1ED514608D1EC  
B4031553E31815C4A259C179F6CD767DCDAE045BF059C4DDA545  
2FDA7C13C54182C5247B853B8EC7F711600B173161C28B9ACA5DF  
9EC2EE09940ECF2DD6AF577DACP5907677C00CD6192DD5A2B79  
39A1FC8C9AFDA458776FED2A3B0196820FBA13464ECE8CC142A  
B7219BD48A806B9B731851D0463C2E90EC2242532330E0C68F82F7  
51C82F5CD98E80DA0AFA63C9537803C3C95CD34A9B607405EE55  
A38590387521DF8CCDCEF33EFFE3409B6B83B78EE681C7335BFC  
7F6FFF96F48433195C09BF06E5F72ECDD625C6E1BDACC608A9E  
DB0384EDE57C65B2A533EB9E0299DBCFABC93E0DA39C1D9EF2  
2C81823FF2E310F0505B33E87AC8747A269CEB56F49DEF3BD92B8  
7F2B72F3A33F99683078926EA386A74EB1B40EF4CC4BCFA3B1354

4F3ED7D723727634CF8D2EB0593AE9EC812479F20556B321856AA9  
63D55E57DA3680E259832A50E63DD97F537CE8951B8A84A2B9D5B  
D4AAA8BDEEA283CAD6BBCE12B37C7FC6BA7F8B035710FB<sup>A</sup>F  
BF5F3274B6F23D4108F8AA023A0494DA716F8F6829C02C1A126235  
1219DD2B936781090C7F5395892E12B2754DF6AB8E86DDB152A0F  
A6987FD2A49C691DFCF448026B369684B9074D8222FDB8F3F8AE  
BED34F87E8D03278796FCBB44A8C5782EDE5DD5CA797009C9EF  
338D4EC5F5205458110F69C787C4608C2DF0017D3ED2A8497D861D  
02FADBF7B0BA225ADEE2EF3D4368F3886FE3EA39258A3A9F0F5  
CB4AFB9BF744045F4DFD80AF21877357EACDC2EA9CD0144603A  
0955A907712BA6F0A2E89257EF0E543488197171CAA3B6D00E5DA  
E0F82256E9F1EA9B9316D2E6CBC3F4B4B4273F9AA108BC5216BE  
289289BA6C6FCC60F8EFDAABF4DEED6F0C126811C9380F7CC29  
B974E371BC492323A8AD74119E8DBDD0D5C751C159A8CE5969EC  
02CF4AFE521B04E774F2F0DC7A73AE969F2BE0BAE7BCAC4B678  
7420055E34051E0DCE87AC965BAF2586E0CB3FDE065E3E0F6088F  
7A4569D85EFBCFBAE7ACD58EDF30C456EF81807757A939AEA0F  
0C6885995E72291A4BA8633A18DCAE58060919D5B4A88D81FDED  
DF0D05825FC3428F22F4EF29FA53A92054F7D3198157F13FFA9A1  
85A5ADF74392B4E97816C843ACCE4F06A3DE99ADDEBEF7BC5B  
12E89FAF39378F0326CED96708FC3F052BD315B0D208C6BE48246  
E7090BECE67D38F29DA26BEE30A4E33539B25E02AAC6DB84AD6  
CAD0C3F89E504D0F04848AC4A14039DE1F8EAD2789093F3CA05A  
5CA799F36DF783C437163D62E6C8D0BDA2DD2E2DF0EC8A77898  
A9789B921DABA4692C5C9D1DD06467468E4118620294C9D1759135  
4B204898695D87C64A3140E0DB1DA31B0D376FEB982FE365EAFD  
465FC64A0C5F8F3F9003489415899D59A543D8208C54A3166529B53  
922F6CAC9A5DC4CD2E48B4EE69E15B42502768E890E31566351B2  
5B2760A41EA89D8626ED79D451140800E03B59B956F8210E5560674  
07D13DC90FA9E8B872BFB8F

ct =EFDBB1F32C9B99E36CC08DA0CF4D5E5C1778C8CF3BD8FC  
EC27041E648BA97BA78B950BE10A2B9637D2AD73AEE68C4C6B7  
8BF45FE77AB726528770F00390F52299D5E78894461C99085B8AE66  
500711A993A0CF0138DC28F1072A64E3E46E85B642FB82887312B8

728C4F00D7FB725EA9F7B5148766292E9CC3EE91F30F62CA9926F  
0142DC476A28CDD48D78645F8BED47630A1A988359FEFD6DECF  
2A3FF5E75914C4C8DE2855787EE9746EE69D1A18C0AE1D28D3E2  
217AD578F61D588DCCABA120E63C9672D372D638C1A3A55DCF6  
351BFDD7B2752DE38D52456816B1DAAD9983E3B0B7CD6C582731  
405BEFD6535F38446B70904840C5D094FED4D972B943E1350793F51  
0AA8C160E3F64926667E660FB5F2C12F43CD3E21B3CFD4953FDF  
7E93A174ADADCB2064C756113A461F0FF6C0D03BA02C09A04303  
FC3F965E0376E554751E947A566EB0036CE537237560113D74C8205  
5E09F104A343A6AAF5BA4901332946FE0C2EA8F28C08FFF5E7BD  
490D8F4F5FD03B0299DB0C0D131B30E878A468F7431798B498E5A4  
6B741AACF6923D7A4822D6A8162409657ABE4382BBCAD50C4327  
3DEA32B1888150BBF56DA18A2D386FD3005D036D363020277DF69  
35CD2923B2C6544AB7A9E2FCC5A352258A9DF14051C616C3226D8  
2EB4C7FA2043380D993F2BB09F6C164B7179E84BDA24AFAC61FF  
F155F63ED1521DDFF6C0B276E7D7995BB726A5F12BC77C69B385  
1ED2FEAE649D327D0B02AC28845F72D6185C410AFC108D05BD38  
7D4A8848F2C3802CE1167CB9088E6936E89D46A2AA4039A5AACE  
03E2892A26D36FFE527A785D2B0EC5AB3B5C197409A61F851963B  
0F07C49112F4DBF2464E0E340D7C72385A2E8F06A4ECFFDD513A  
D17F66322333D5F1C559BCD0F2BE2754FC3472525D30615D9E71D  
514275CBAD506B4831915A0AAE90E054C39100F9116B6FE65ADE6  
34167EF48B46F6CD7096B4313966D84FD3E11BB66FABADB7CCF1  
D62541F33A594325DF59D32CFD1490933AF9BC5B667205516C7F7B  
BB665F0382FAE396B70DF864387F151F3CAF2DA785DA0913890E  
C3DD8CD4D6036B10677E1118E8689C063762D694D8EC2501DE18C  
803A6C7EA0D8E4535EBEE07EFC6338CF3E5D0559DACF60C65AF  
11EC264F2720DFC82D5EE1A60781FB825F0B970E3595C31833B6F8  
2FB17531960AA66B0705C302B85EA4FDB82F06C20186C1DFC9599  
8297C1B7C413CC7AE6F02427B53F6DC98C8D4229A55B6F16F3160  
9DBEFEB2BAC57F517F11E063FF78801FE437526D66B388D99EBD  
DD5513CB4EB884BF2E8CC567AACF96A371A182C21B280E6351E  
FC949F1F20A078EDAAFE24C0DCA4E97D0604C1D7FCECC3FE30  
5810AE6C23200F8C7682AF35C5C3B53C78B0DB4BC17234E

ss = DAFEE7F565080D7998D99D5C0558E58CAAD0F9AE55BC4614  
AB4D53C3FE5FB452

## 5 可证明安全

### 5.1 困难假设

**LWE问题.** 令  $n, q$  是任意正整数,  $\alpha$  是任意正实数,  $\chi_\alpha$  是  $\mathbb{Z}$  上以  $\alpha$  为参数的离散分布  $\chi_\alpha \subseteq \mathbb{Z}$ 。对于向量  $\mathbf{s} \in \mathbb{Z}_q^n$ , 定义分布  $A_{\mathbf{s}, \alpha}$  如下:

$$A_{\mathbf{s}, \alpha} = \{(\mathbf{a}, b = \mathbf{a}^T \mathbf{s} + e \bmod q) : \mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n, e \xleftarrow{\$} \chi_\alpha\}.$$

当随机均匀地选取秘密向量  $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$  时, 计算性 LWE 问题的目标是在给定分布  $A_{\mathbf{s}, \alpha}$  中任意多项式个样本的条件下计算出秘密向量  $\mathbf{s} \in \mathbb{Z}_q^n$ 。而对应的判定性 LWE 问题的目标则是在给定任意多项式个样本的条件下区分分布  $A_{\mathbf{s}, \alpha}$  和  $\mathbb{Z}_q^n \times \mathbb{Z}_q$  上的均匀分布。对于满足某些条件的  $q$  和分布, 判定性 LWE 问题在平均情况下的困难性在多项式时间的意义下等价于计算性 LWE 问题在最坏情况下的困难性[35,33,7]。当分布  $\chi_\alpha$  是以  $\alpha$  为标准差的高斯分布时, Regev 证明了相应的  $\text{LWE}_{n, q, \alpha}$  问题在平均情况下的困难性可以量子归约到格上某些问题在最坏情况下的困难性[35]。此后, 对于某些特定的参数, Peikert[33]给出了  $\text{LWE}_{n, q, \alpha}$  到格上困难问题的经典归约。特别地, 结合文献[35,34,23]的结论, 我们有如下命题成立:

**命题 1** 令实数  $\alpha = \alpha(n) \in (0, 1)$  和素数  $q = q(n)$  满足条件  $\alpha q > 2\sqrt{n}$ 。如果存在多项式时间的(量子)算法求解  $\text{LWE}_{n, q, \alpha q \sqrt{2}}$  问题, 那么存在多项式时间的量子算法求解秩为  $n$  的格上近似因子为  $\gamma = \tilde{O}(n/\alpha)$  的最坏情况下的  $\text{SIVP}_\gamma$  问题。

由  $\text{SIVP}_\gamma$  的困难性可知, 对于任意常数  $\epsilon < 1/2$  和实数  $\alpha = 2^{-n^\epsilon}$ ,  $\text{LWE}_{n, q, \alpha q \sqrt{2}}$  仍然是困难的。此外, 当秘密元素  $\mathbf{s}$  并不是随机均匀地选自于  $\mathbb{Z}_q^n$  时, LWE 问题也可能是非常困难的。特别地, 当  $\mathbf{s} \xleftarrow{\$} \chi_\alpha^n$  时, 相应的  $\text{LWE}_{n, q, \alpha}$  问题至少和标准的 LWE 问题是一样困难的[7,28]。事实上, 这类变种的问题称为 LWE 的正规形, 在全同态加密中常常被用来控制错误项的增长[15,12]。

**RLWE问题.** 令整数  $n$  是 2 的幂次, 素数  $q$  满足  $q = 1 \bmod 2n$ , 环  $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ 。对于任意环元素  $s \in R_q$  和实数  $\alpha$ , 定义分布  $B_{s, \alpha}$  如下:

$$B_{s, \alpha} = \{(a, b = as + e) : a \xleftarrow{\$} R_q, e \xleftarrow{\$} \chi_\alpha^n\}.$$

对于随机均匀选取地秘密元素  $s \xleftarrow{\$} R_q$  和任意多项式界定的正整数  $\ell$ , RLWE 问题  $\text{RLWE}_{n, q, \ell, \alpha}$  的目标是在给定  $\ell$  个样本的条件下区分分布  $B_{s, \alpha}$  和  $R_q \times R_q$  上的均匀分布。对于适当选取的参数, 我们有如下结论成立:

**命题 2 ([28, 定理3.6])** 令正整数  $n$  是 2 的幂次,  $\alpha \in (0, 1)$  是一个实数,  $q$  是一个素数, 定义  $R = \mathbb{Z}[x]/(x^n + 1)$ 。如果  $q = 1 \bmod 2n$  且  $\beta q > \omega(\sqrt{\log n})$ , 那么存在从环  $R$  的理想格上最坏情况的  $\text{SIVP}_{\tilde{O}(\sqrt{n}/\beta)}$  问题到平均情况的  $\text{RLWE}_{n,q,\ell,\alpha}$  问题的多项式时间的量子归约, 其中  $\alpha = \beta q \cdot (n\ell / \log(n\ell))^{1/4}$ 。

类似地, 当秘密元素  $s \leftarrow \chi_\alpha^n$  时, 相应的  $\text{RLWE}_{n,q,\alpha,\ell}$  问题至少和环上标准的  $\text{RLWE}$  问题是一样困难的 [7, 28]。

**MLWE问题.**  $\text{RLWE}$  具有更好的结构, 从而使得基于  $\text{RLWE}$  问题设计的密码方案无论是运行速度还是密钥/密文大小都具有较好的效率表现, 但参数选择往往比较受限; 而  $\text{LWE}$  问题的参数选择则具有较高的灵活性。综合考虑安全性和效率, 文献 [12, 1] 提出了标准  $\text{LWE}$  问题和  $\text{RLWE}$  问题的结合版本—模  $\text{LWE}$  问题 (MLWE)。特别地, 令整数  $n$  是 2 的幂次, 素数  $q$  满足  $q = 1 \bmod 2n$ , 环  $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ 。对于任意正整数  $k, \ell \geq 1$ , 正实数  $\alpha$ , 判定性  $\text{MLWE}$  问题  $\text{MLWE}_{n,q,k,\ell,\alpha}$  的目标是将样本  $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$  和选自于  $R_q^{k \times \ell} \times R_q^k$  上均匀分布的元组区分开, 其中  $\mathbf{A} \leftarrow R_q^{k \times \ell}$ ,  $\mathbf{s} \leftarrow (\chi_\alpha^n)^\ell$ ,  $\mathbf{e} \leftarrow (\chi_\alpha^n)^k$ 。显然, 当  $R_q = \mathbb{Z}_q$  (即  $n = 1$ ) 时,  $\text{MLWE}$  问题就是标准的  $\text{LWE}$  问题, 而当  $\ell = 1$  时,  $\text{MLWE}$  问题就是标准的  $\text{RLWE}$  问题。因此, 研究者们倾向于相信  $\text{MLWE}$  问题的困难性介于标准  $\text{LWE}$  问题与  $\text{RLWE}$  问题之间。特别地, 目前所有求解  $\text{LWE}$  和  $\text{RLWE}$  问题的算法在求解  $\text{MLWE}$  时并没有明显的优势, 事实上, 目前最好的求解算法都没有用到  $\text{RLWE}$  和  $\text{MLWE}$  问题的环结构。

**ALWE问题** 令  $n, q$  是任意正整数,  $\alpha_1, \alpha_2$  是任意正实数,  $\chi_{\alpha_1}, \chi_{\alpha_2}$  是  $\mathbb{Z}$  上以  $\alpha_1, \alpha_2$  为参数的离散分布。对于向量  $\mathbf{s} \leftarrow \chi_{\alpha_1}^n$ , 定义非对称  $\text{LWE}$  (ALWE) 分布  $A_{\mathbf{s}, \alpha_2}$  如下:

$$A_{\mathbf{s}, \alpha_2} = \{(\mathbf{a}, b = \mathbf{a}^T \mathbf{s} + e \bmod q) : \mathbf{a} \leftarrow \mathbb{Z}_q^n, e \leftarrow \chi_{\alpha_2}\}.$$

其中非对称  $\text{LWE}$  是由正规形  $\text{LWE}$  演变而来。在正规形  $\text{LWE}$  中, 向量  $\mathbf{s}$  的各分量和错误  $e$  取自同一分布, 而在非对称  $\text{LWE}$  分布中, 秘密向量  $\mathbf{s}$  各分量和错误  $e$  可取自参数不同的 (相同) 分布。判定性  $\text{ALWE}$  问题  $\text{ALWE}_{n,q,\ell,\alpha_1,\alpha_2}$  是指在  $\mathbf{s} \in \chi_{\alpha_1}^n$  给定且有  $\ell$  个样本的条件下区分分布  $A_{\mathbf{s}, \alpha_2}$  和  $\mathbb{Z}_q^n \times \mathbb{Z}_q$  上的均匀分布。计算性  $\text{ALWE}$  问题  $\text{ALWE}_{n,q,\ell,\alpha_1,\alpha_2}$  则是指在  $\ell$  个样本的条件下计算出秘密向量  $\mathbf{s} \in \chi_{\alpha_1}^n$ 。

**AMLWE问题.** 结合密码方案的设计和目前针对 (M/R)  $\text{LWE}$  求解算法的现状, 我们提出非对称  $\text{AMLWE}$  问题 (类似地还可以定义  $\text{ARLWE}$  问题)。特别地, 令整数  $n$  是 2 的幂次, 素数  $q$  满足  $q = 1 \bmod 2n$ , 环  $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ 。设  $k, \ell \geq 1$  为正整数,  $\alpha_1, \alpha_2$  为正实数。判定性  $\text{AMLWE}$  问题  $\text{AMLWE}_{n,q,k,\ell,\alpha_1,\alpha_2}$  的目标是将样本  $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$  和选自于  $R_q^{k \times \ell} \times R_q^k$  上均匀分布的元组区分开, 其中  $\mathbf{A} \leftarrow$

$R_q^{k \times \ell}, \mathbf{s} \xleftarrow{\$} (\chi_{\alpha_1}^n)^\ell, \mathbf{e} \xleftarrow{\$} (\chi_{\alpha_2}^n)^k$ 。计算性 AMLWE 问题  $\text{AMLWE}_{n,q,k,\ell,\alpha_1,\alpha_2}$  的目标是给定样本  $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}) \in R_q^{k \times \ell} \times R_q^k$ , 输出环向量  $\mathbf{s} \in R_q^\ell$ , 其中  $\mathbf{A} \xleftarrow{\$} R_q^{k \times \ell}, \mathbf{s} \xleftarrow{\$} (\chi_{\alpha_1}^n)^\ell, \mathbf{e} \xleftarrow{\$} (\chi_{\alpha_2}^n)^k$ 。显然, 当  $\alpha_1 = \alpha_2$  时, AMLWE 问题就退化为标准的 MLWE 问题。此外, 当  $\chi_\alpha$  为高斯分布时, 我们还可以利用高斯分布的性质证明如下困难关系在多项式时间的意义下成立:

$$\text{MLWE}_{n,q,k,\ell,\min(\alpha_1,\alpha_2)} \leq \text{AMLWE}_{n,q,k,\ell,\alpha_1,\alpha_2} \leq \text{MLWE}_{n,q,k,\ell,\max(\alpha_1,\alpha_2)}.$$

换句话说, 只要选择合适的参数, 我们总能够保证 AMLWE 问题是难于求解的。在第6节中, 我们将考虑 MLWE 问题的已有攻击算法及它们的变形来估计 AMLWE 问题的具体求解复杂度。

与文献[1]类似, 由于压缩公钥的需要, 我们还要用到 AMLWE 的一个变种问题, 即 AMLWE-R 问题。特别地, 判定性 AMLWE-R 问题  $\text{AMLWE-R}_{n,q,p,k,\ell,\alpha_1,\alpha_2}$  的目标是将样本

$$(\mathbf{A}, \bar{\mathbf{t}} = \lceil \mathbf{t} \rceil_{q \rightarrow p}, \mathbf{A}\mathbf{s} + \mathbf{e}, \lceil \bar{\mathbf{t}} \rceil_{p \rightarrow q}\mathbf{s} + \mathbf{e})$$

和随机选取的元组  $(\mathbf{A}', \lceil \mathbf{t}' \rceil_{q \rightarrow p}, \mathbf{u}, v) \in R_q^{k \times \ell} \times R_p^\ell \times R_q^k \times R_q$  区分开, 其中  $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} R_q^{k \times \ell}, \mathbf{s} \xleftarrow{\$} (\chi_{\alpha_1}^n)^\ell, \mathbf{e} \xleftarrow{\$} (\chi_{\alpha_2}^n)^k, \mathbf{t}, \mathbf{t}' \xleftarrow{\$} R_q^\ell, \mathbf{u} \xleftarrow{\$} R_q^k, v \xleftarrow{\$} R_q$ 。

为了便于系统的实现, 我们将使用二项分布来作为 AMLWE 的噪音分布。以正整数  $\eta$  为参数的中心二项分布  $B_\eta$  定义如下:

$$B_\eta = \left\{ \sum_{i=1}^{\eta} (a_i - b_i) : (a_1, \dots, a_\eta, b_1, \dots, b_\eta) \xleftarrow{\$} \{0, 1\}^{2\eta} \right\}$$

从  $B_\eta$  中采样一个多项式  $f \in R_q$  或多项式向量的意思是从  $B_\eta$  中采样每个多项式的系数。易证, 以  $\eta$  为参数的二项分布是以  $\sqrt{\eta/2}$  为标准差的亚高斯分布。在一定参数下, 使用二项分布作为错误分布的计算性 AMLWE 问题可以归约到使用高斯分布的 AMLWE 问题。在没有特别说明的情况下, 本文档将使用符号  $\text{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$  和  $\text{AMLWE-R}_{n,q,k,\ell,\eta_1,\eta_2}$  来表示使用二项分布  $B_{\eta_1}$  作为秘密向量  $\mathbf{s}$  的分布和  $B_{\eta_2}$  作为噪音向量  $\mathbf{e}$  的分布的 AMLWE 问题和 AMLWE-R 问题。

**定义 1 (AMLWE 困难假设)** 对于适当选取的正整数  $n, q, k, \ell, \eta_1, \eta_2 \in \mathbb{Z}$ , 不存在 (量子) 多项式时间的敌手能够解决 AMLWE 问题  $\text{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ 。

**定义 2 (AMLWE-R 困难假设)** 对于适当选取的正整数  $n, q, p, k, \ell, \eta_1, \eta_2 \in \mathbb{Z}$ , 不存在 (量子) 多项式时间的敌手能够解决 AMLWE-R 问题  $\text{AMLWE-R}_{n,q,p,k,\ell,\eta_1,\eta_2}$ 。

附录A.1 和 A.2 给出了公钥加密方案选择明文攻击安全性 (即 IND-CPA 安全性) 和密钥封装机制选择密文攻击安全性的定义 (即 IND-CCA 安全性)。接下来

两个小节中，我们将在（量子）随机预言机模型分别证明埃奎斯公钥加密方案和埃奎斯密钥封装机制在以上两个困难假设下分别达到了IND-CPA安全性和IND-CCA安全性。特别地，在随机预言机模型(ROM)[9]中，敌手 $\mathcal{A}$ 能询问一个随机预言机多项式次。在量子随机预言机模型(QROM)[10]中，敌手 $\mathcal{A}$ 能用任意输入字符串构成的量子叠加态(Superpositions)作为输入来询问量子随机预言机，且可以执行任意多项式次这样的询问。

## 5.2 埃奎斯公钥加密方案的选择明文攻击安全性

我们证明埃奎斯公钥加密方案在AMLWE和AMLWE-R困难假设下是IND-CPA安全的。为了证明的更加直观和简洁，我们忽略相关用于实现的编码和解码，以及NTT等相关操作，而主要关注于下列形式化方案的设计。正式地，令 $n, k, q, \eta_1, \eta_2, d_u, d_v, d_t$ 与第3.1中的含义相同，令 $\mathcal{H} : \mathcal{B}^{n/8} \rightarrow R_q^{k \times k}$ 是用于生成公钥中矩阵 $\mathbf{A}$ 的杂凑函数，那么我们可以将第3.1中的公钥加密方案形式化的描述为 $\text{Aigis-pke}' = (\text{Aigis-pke}'.\text{KeyGen}, \text{Aigis-pke}'.\text{Enc}, \text{Aigis-pke}'.\text{Dec})$ 。特别地，如果在实现中用于生成矩阵 $\mathbf{A}$ 的Parse和XOF函数的组合满足随机预言机的性质， $\mathbf{G} : \mathcal{B}^{n/8} \rightarrow \mathcal{B}^{n/4}$ 是伪随机生成器，并且 $\text{PRF} : \mathcal{B}^{n/8} \times \mathbb{Z} \rightarrow \mathcal{B}^{n \cdot \max(\eta_1, \eta_2)/4}$ 是伪随机函数，那么公钥加密方案 $\text{Aigis-pke}'$ 与第3.1节中描述的公钥加密方案的安全性相同。正式地，

- **密钥生成算法** $\text{Aigis-pke}'.\text{KeyGen}(\kappa)$ : 随机选择 $\rho \xleftarrow{\$} \mathcal{B}^{n/8}$ ,  $\mathbf{s} \xleftarrow{\$} B_{\eta_1}^k$ ,  $\mathbf{e} \xleftarrow{\$} B_{\eta_2}^k$ , 计算 $\mathbf{A} = \mathcal{H}(\rho) \in R_q^{k \times k}$ 和 $\mathbf{t} = \mathbf{As} + \mathbf{e} \in R_q^k$ ,  $\bar{\mathbf{t}} = \lceil \mathbf{t} \rceil_{q \rightarrow 2^{d_t}}$ , 最后返回公钥 $pk = (\rho, \bar{\mathbf{t}})$ 和私钥 $sk = \mathbf{s}$ 。
- **加密算法** $\text{Aigis-pke}'.\text{Enc}(pk, \mu)$ : 给定公钥 $pk = (\rho, \bar{\mathbf{t}})$ 和明文消息 $\mu \in R_2$ , 随机选择 $\mathbf{r} \xleftarrow{\$} B_{\eta_1}^k$ ,  $\mathbf{e}_1 \xleftarrow{\$} B_{\eta_2}^k$ ,  $e_2 \xleftarrow{\$} B_{\eta_2}$ , 计算 $\mathbf{A} = \mathcal{H}(\rho)$ ,  $\mathbf{u} = \mathbf{A}^T \mathbf{r} + \mathbf{e}_1$ ,  $v = \lceil \bar{\mathbf{t}} \rceil_{2^{d_t} \rightarrow q}^T \mathbf{r} + e_2$ , 最终输出密文 $c = (\bar{\mathbf{u}} = \lceil \mathbf{u} \rceil_{q \rightarrow 2^{d_u}}, \bar{v} = \lceil v + \mu \cdot \lceil \frac{q}{2} \rceil \rceil_{q \rightarrow 2^{d_v}})$ 。
- **解密算法** $\text{Aigis-pke}'.\text{Dec}(sk, c)$ : 给定私钥 $sk = \mathbf{s}$ 和密文 $c = (\bar{\mathbf{u}}, \bar{v})$ , 计算 $z = \lceil \bar{v} \rceil_{2^{d_v} \rightarrow q} - \mathbf{s}^T \lceil \bar{\mathbf{u}} \rceil_{2^{d_u} \rightarrow q}$ , 输出 $\lceil z \rceil_{q \rightarrow 2} = \lceil z \cdot \frac{2}{q} \rceil \bmod 2$ 。

**定理 1** 令参数 $n, q, k, \eta_1, \eta_2, d_u, d_v, d_t$ 与埃奎斯公钥加密方案的取值相同。如果函数 $\mathcal{H} : \mathcal{B}^{n/8} \rightarrow R_q^{n \times n}$ 是随机预言机，那么基于 $\text{AMLWE}_{n, q, k, k, \eta_1, \eta_2}$ 困难假设和 $\text{AMLWE-R}_{n, q, 2^{d_t}, k, k, \eta_1, \eta_2}$ 困难假设，方案 $\text{Aigis-pke}'$ 是可证明选择明文攻击安全的。

**证明.** 对于任意多项式时间的敌手 $\mathcal{A}$ ，令 $\mathcal{A}$ 赢得方案 $\text{Aigis-pke}'$ 选择明文攻击安全实验的优势为 $\epsilon_A$ 。接下来，我们将通过实验 $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2$ 来证明  $\epsilon_A \leq \text{negl}(\kappa)$ ，从而完成定理1的证明。

**实验 $\mathbf{G}_0$ :** 实验 $\mathbf{G}_0$ 是真实的IND-CPA安全实验。正式地，该实验将按如下方式为敌手模拟攻击环境：

- 随机预言机 $\mathcal{H}$ 的模拟：该实验将为随机预言机 $\mathcal{H}$ 维护一个询问列表 $\mathcal{L} := \{(\rho_i, \mathbf{A}_i)\}$ 。当收到敌手 $\mathcal{A}$ 的随机预言机询问 $\rho$ 时，实验首先查询列表 $\mathcal{L}$ 中是否存在元组 $(\rho, \mathbf{A})$ 。如果不存在，随机选择 $\mathbf{A} \xleftarrow{\$} R_q^{k \times k}$ 并将 $(\rho, \mathbf{A})$ 加入到列表 $\mathcal{L}$ 。然后，将矩阵 $\mathbf{A}$ 返回给敌手 $\mathcal{A}$ ；
- 生成公钥：运行密钥生成算法 $(pk, sk) \leftarrow \text{Aigis'}.KeyGen(\kappa)$ ，并将公钥 $pk = (\rho, \bar{\mathbf{t}})$ 交给敌手 $\mathcal{A}$ ；
- 生成挑战密文：收到 $\mathcal{A}$ 的挑战明文 $\mu_0, \mu_1 \xleftarrow{\$} R_2$ 后，随机选择 $b \xleftarrow{\$} \{0, 1\}$ ，计算 $c \leftarrow \text{Aigis'}.Enc(pk, \mu_b)$ ，并将密文 $c = (\bar{\mathbf{u}}, \bar{v})$ 发送给敌手 $\mathcal{A}$ 。

最后，敌手 $\mathcal{A}$ 将输出对 $b \in \{0, 1\}$ 的猜测 $b' \in \{0, 1\}$ 。

令 $\text{Adv}(\mathbf{G}_i) = \Pr[b' = b] - 1/2$ 表示在实验 $\mathbf{G}_i$ 中敌手猜中 $b$ 的优势。根据我们随机预言机模型的假设和相关定义，我们有 $\text{Adv}(\mathbf{G}_0) = \epsilon_A$ 。

**实验 $\mathbf{G}_1$ ：**除了在 $\text{Aigis-pke'}.KeyGen$ 算法中直接随机选择 $\mathbf{A} \xleftarrow{\$} R_q^{k \times k}$ 和 $\mathbf{t} \xleftarrow{\$} R_q^k$ 来生成公钥 $pk$ 之外，实验 $\mathbf{G}_1$ 与 $\mathbf{G}_0$ 相同。

如果存在敌手 $\mathcal{A}$ 能够区分实验 $\mathbf{G}_1$ 与实验 $\mathbf{G}_0$ ，那么我们可以构造一个攻击AMLWE困难假设的PPT敌手 $\mathcal{C}_1$ 。具体地，给定 $\text{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ 问题的一个实例 $(\mathbf{A}, \mathbf{t}) \in R_q^{k \times k} \times R_q^k$ 作为输入，敌手 $\mathcal{C}_1$ 要判断 $(\mathbf{A}, \mathbf{t})$ 是否选自于 $R_q^{k \times k} \times R_q^k$ 的均匀随机分布。正式地， $\mathcal{C}_1$ 将按下列方式修改公钥 $pk$ 的生成方式，除此之外， $\mathcal{C}_1$ 与敌手 $\mathcal{A}$ 的交互与实验 $\mathbf{G}_0$ 中相同：

- 生成公钥：随机选择 $\rho \xleftarrow{\$} \mathcal{B}^{n/8}$ ，查询列表 $\mathcal{L}$ 中是否存在元组 $(\rho, *)$ 。如果存在，则终止实验。否则，将元组 $(\rho, \mathbf{A})$ 加入列表 $\mathcal{L}$ （即定义 $\mathcal{H}(\rho) = \mathbf{A}$ ）。计算 $\bar{\mathbf{t}} = \lceil \mathbf{t} \rceil_{q \rightarrow 2^{d_t}}$ ，并输出公钥 $pk = (\rho, \bar{\mathbf{t}})$ 给敌手。

最后， $\mathcal{C}_1$ 将 $\mathcal{A}$ 的猜测 $b' \in \{0, 1\}$ 作为自己对于 $\text{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ 问题的解。

首先，由 $\mathcal{H}$ 是随机预言机的假设和 $\rho \in \mathcal{B}^{n/8}$ 的随机性，列表 $\mathcal{L}$ 中是否存在元组 $(\rho, *)$ 的概率是可忽略的。其次，如果 $(\mathbf{A}, \mathbf{t})$ 随机选自于 $R_q^{k \times k} \times R_q^k$ ，那么 $\mathcal{C}_1$ 与敌手 $\mathcal{A}$ 的交互与实验 $\mathbf{G}_1$ 中相同，否则 $\mathcal{C}_1$ 与敌手 $\mathcal{A}$ 的交互与实验 $\mathbf{G}_0$ 中相同。换句话说，如果敌手 $\mathcal{A}$ 能够区分实验 $\mathbf{G}_0$ 和 $\mathbf{G}_1$ ，那么 $\mathcal{C}_1$ 能够解决 $\text{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ 问题。因此，在 $\text{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ 困难假设下，我们有如下结论：

$$|\text{Adv}(\mathbf{G}_1) - \text{Adv}(\mathbf{G}_0)| \leq \text{negl}(\kappa)$$

**实验 $\mathbf{G}_2$ ：**除了在 $\text{Aigis-pke'}.Enc$ 算法中直接随机选择 $\mathbf{u} \xleftarrow{\$} R_q^k$ 和 $v \in R_q$ 生成挑战密文之外，实验 $\mathbf{G}_2$ 与 $\mathbf{G}_1$ 相同。

如果存在敌手 $\mathcal{A}$ 能够区分实验 $\mathbf{G}_2$ 与实验 $\mathbf{G}_1$ ，那么我们可以构造一个攻击AMLWE-R困难假设的PPT敌手 $\mathcal{C}_2$ 。具体地，给定 $\text{AMLWE-R}_{n,q,k,k,\eta_1,\eta_2}$ 问题的一个实例 $(\mathbf{A}, \lceil \mathbf{t} \rceil_{q \rightarrow 2^{d_t}}, \mathbf{u}, v) \in R_q^{k \times k} \times R_{2^{d_t}}^k \times R_q^k \times R_q$ 作为输入，敌手 $\mathcal{C}_2$ 要判断 $(\mathbf{A}, \mathbf{t}, \mathbf{u}, v)$ 是否选自于 $R_q^{k \times k} \times R_q^k \times R_q^k \times R_q$ 的随机均匀分布。正式地， $\mathcal{C}_2$ 将按下列方式与敌手 $\mathcal{A}$ 的交互：

- **生成公钥:** 随机选择  $\rho \xleftarrow{\$} \mathcal{B}^{n/8}$ , 查询列表  $\mathcal{L}$  中是否存在元组  $(\rho, *)$ 。如果存在, 则终止实验。否则, 将元组  $(\rho, \mathbf{A}^T)$  加入列表  $\mathcal{L}$  (即定义  $\mathcal{H}(\rho) = \mathbf{A}^T$ )。令  $\bar{\mathbf{t}} = \lceil \mathbf{t} \rceil_{q \rightarrow 2^{d_t}}$ , 并输出公钥  $pk = (\rho, \bar{\mathbf{t}})$  给敌手  $\mathcal{A}$ ;
- **生成挑战密文:** 收到  $\mathcal{A}$  的挑战明文  $\mu_0, \mu_1 \xleftarrow{\$} R_2$  后, 随机选择  $b \xleftarrow{\$} \{0, 1\}$ , 计算  $\bar{\mathbf{u}} = \lceil \mathbf{u} \rceil_{q \rightarrow 2^{d_u}}, \bar{v} = \lceil v + \mu_b \cdot \lceil \frac{q}{2} \rceil \rceil_{q \rightarrow 2^{d_v}}$ , 并将密文  $c = (\bar{\mathbf{u}}, \bar{v})$  发送给  $\mathcal{A}$ 。

最后,  $\mathcal{C}_1$  将  $\mathcal{A}$  的猜测  $b' \in \{0, 1\}$  作为自己对于 AMLWE-R <sub>$n, q, k, k, \eta_1, \eta_2$</sub>  问题的解。

显然, 如果  $(\mathbf{A}, \mathbf{t}, \mathbf{u}, v)$  随机选自于  $R_q^{k \times k} \times R_q^k \times R_q^k \times R_q$ , 那么  $\mathcal{C}_1$  与敌手  $\mathcal{A}$  的交互与实验  $\mathbf{G}_1$  相同, 否则  $\mathcal{C}_2$  与敌手  $\mathcal{A}$  的交互与实验  $\mathbf{G}_2$  中相同。换句话说, 如果敌手  $\mathcal{A}$  能够区分实验  $\mathbf{G}_2$  和  $\mathbf{G}_1$ , 那么  $\mathcal{C}_1$  能够解决 AMLWE-R <sub>$n, q, 2^{d_t}, k, k, \eta_1, \eta_2$</sub>  问题。因此, 在 AMLWE-R <sub>$n, q, 2^{d_t}, k, k, \eta_1, \eta_2$</sub>  困难假设下, 我们有如下结论:

$$|\text{Adv}(\mathbf{G}_2) - \text{Adv}(\mathbf{G}_1)| \leq \text{negl}(\kappa)$$

由于在实验  $\mathbf{G}_2$  中明文  $\mu_b$  被随机选择的  $v$  完美隐藏, 因此我们有  $\text{Adv}(\mathbf{G}_2) = 0$ 。综上所述, 我们有  $\epsilon_A \leq \text{negl}(\kappa)$ , 从而定理 1 得证。  $\square$

### 5.3 埃奎斯密钥封装机制的选择密文攻击安全性

由于埃奎斯密钥封装机制是通过将 Fujisaki-Okamoto (FO) 变换 [24, 21] 作用到选择明文安全的埃奎斯公钥加密方案而得到的, 结合文献 [24, 11] 和定理 1 中的结论, 我们有如下定理:

**定理 2** 基于 AMLWE 困难假设和 AMLWE-R 困难假设, 埃奎斯密钥封装机制在随机预言机模型下是可证明选择密文攻击安全的。

进一步, 文献 [25] 证明了如果底层公钥加密方案是选择明文攻击安全的, 那么由 FO 变换得到的隐式拒绝密钥封装机制在量子预言机模型下是选择密文攻击安全的。注意到, 当解封装失败时, 埃奎斯密钥封装机制仍然会返回一个随机的“密钥”(即隐式拒绝)。结合文献 [25] 和定理 1 中的结论, 我们有如下定理:

**定理 3** 基于 AMLWE 困难假设和 AMLWE-R 困难假设, 埃奎斯密钥封装机制是在量子随机预言机模型下是可证明选择密文攻击安全的。

## 6 抵抗已知攻击的能力

求解 LWE 的算法主要包括原始攻击 (primal attack), 对偶攻击 (dual Attack) 以及利用 BKW、Arora-Ge 方法直接求解 [5] 等。由于 BKW、Arora-Ge 攻击方法需要的样本数据量多为指数或次指数量级, 这两类方法并不适用于分析只有比

较少样本的具体密码方案。因此，对实际格上密码系统的分析往往只考虑原始攻击和对偶攻击这两种目前最有效的攻击方法。此外，由于这两类方法在求解RLWE和MLWE问题时并没有比求解标准LWE问题更有优势，因此文献中在分析基于MLWE和RLWE问题密码方案的具体安全强度时，往往只是将相应的RLWE或MLWE问题转换成标准LWE问题来进行分析[19,11]。特别地，我们首先会将 $\text{AMLWE}_{n,q,k,\ell,\alpha_1,\alpha_2}$ 问题转换成 $\text{ALWE}_{nk,q,k\ell,\alpha_1,\alpha_2}$ 问题，然后再将针对LWE问题的求解方法推广到求解ALWE问题。由于任意其他有界中心对称分布都可以看成服从一定参数的亚高斯分布，为了便于分析且不失一般性，我们将只考虑秘密向量 $\mathbf{s} \xleftarrow{\$} \chi_{\alpha_1}^n$ 和噪音向量 $\mathbf{e} \xleftarrow{\$} \chi_{\alpha_2}^m$ 的各分量分别选自于以 $\alpha_1$ 和 $\alpha_2$ 为标准差的亚高斯分布的 $\text{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ 问题。具体地，我们的目标是在给定问题样本

$$(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$$

的情况下，计算并输出 $\mathbf{s} \in \mathbb{Z}_q^n$ ，其中 $\mathbf{s} \xleftarrow{\$} \chi_{\alpha_1}^n$ ,  $\mathbf{e} \xleftarrow{\$} \chi_{\alpha_2}^m$ 。

## 6.1 原始攻击及其变形

原始攻击的基本思路是通过嵌入的方式将ALWE问题转化为求解适当的格上有界译码问题(BDD)或短向量问题，不同原始攻击的主要区别在于嵌入的方式不同。我们将考虑以下针对ALWE问题 $\text{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ 的原始攻击及其变形。

**传统原始攻击.** 传统原始攻击利用了Kannan嵌入[26,4]将LWE的求解问题转换成求解格中的唯一最短向量问题(uSVP)。首先，定义格

$$\Lambda = \{\mathbf{y} \in \mathbb{Z}^m | \mathbf{y} = \mathbf{Ax} \bmod q, \mathbf{x} \in \mathbb{Z}_q^n\},$$

则格 $\Lambda$ 的维数 $d = m$ 。当 $m$ 足够大于 $n$ 时，矩阵 $\mathbf{A}$ 以很大概率存在 $n$ 个线性无关的行。不妨设 $\mathbf{A}$ 的前 $n$ 行线性无关（否则我们可以通过行变换将线性无关的行置换到前 $n$ 行），且设前 $n$ 行对应的子矩阵为 $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ ，即  $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix}$ 。记矩阵 $\mathbf{A}_1^{-1} \in \mathbb{Z}_q^{n \times n}$ 为 $\mathbf{A}_1$ 的逆，令  $\mathbf{A}' = \begin{pmatrix} \mathbf{I}_n \\ \mathbf{A}_2 \mathbf{A}_1^{-1} \end{pmatrix}$ ，则我们有

$$\Lambda = \{\mathbf{y} | \mathbf{y} = \mathbf{Ax} \bmod q, \mathbf{x} \in \mathbb{Z}_q^n\} = \{\mathbf{y} | \mathbf{y} = \mathbf{A}'\mathbf{x} \bmod q, \mathbf{x} \in \mathbb{Z}_q^n\} \subset \mathbb{Z}^m,$$

且以下矩阵

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_n & 0 \\ \mathbf{A}_2 \mathbf{A}_1^{-1} & q\mathbf{I}_{m-n} \end{pmatrix} \in \mathbb{Z}^{m \times m}$$

的列向量构成格 $\Lambda$ 的一组基，容易有 $\det(\Lambda) = q^{m-n}$ 。由 $\mathbf{b} = \mathbf{As} + \mathbf{e} \bmod q$ 可知向量 $\mathbf{b}$ 距离格 $\Lambda$ 的距离为 $\|\mathbf{e}\|$ 。如果能找到一个离 $\mathbf{b}$ 最近的格点 $\mathbf{u} \in \Lambda$ ，则我们将

以很大概率有  $\mathbf{e} = \mathbf{b} - \mathbf{u}$ 。换句话说，求解向量  $\mathbf{e}$  的问题可转化为格  $\Lambda$  上的有界译码问题(BDD)，从而可利用文献[27]中的多最近平面算法进行求解。进一步，利用Kannan嵌入[26,4]可以将BDD问题转化为唯一最短向量问题(uSVP)。具体地，考虑由以下矩阵  $\mathbf{B}'$  的列向量生成的格  $\Lambda'$ ，

$$\mathbf{B}' = \begin{pmatrix} \mathbf{I}_n & 0 & \mathbf{b} \\ \mathbf{A}_2 \mathbf{A}_1^{-1} & q\mathbf{I}_{m-n} & 0 \\ 0 & 0 & t \end{pmatrix} \in \mathbb{Z}^{(m+1) \times (m+1)},$$

其中  $t \in \mathbb{Z}$  是一个可调参数。理论上，取  $t = \|\mathbf{e}\|$  时能够取得较好的效果，但实际实验则显示  $t = 1$  时效果较好，所以我们通常选取  $t = 1$ 。在这种情况下，我们以极大的概率有  $\mathbf{v} = (\mathbf{e}^T, 1)^T \in \mathbb{Z}^{m+1}$  为格  $\Lambda'$  中唯一最短向量。因此，我们可以通过求解格  $\Lambda'$  中 uSVP 问题来求解噪音向量  $\mathbf{e} \in \mathbb{Z}^m$ ，进而通过求解线性方程组来恢复私钥  $\mathbf{s} \in \mathbb{Z}_q^n$ ，即计算  $\mathbf{As} = \mathbf{b} - \mathbf{e}$ 。此外，由于向量  $\mathbf{e}$  的每个分量都选自于亚高斯分布  $\chi_{\alpha_2}$ ，我们有  $\|\mathbf{v}\| \approx \|\mathbf{e}\| \approx \alpha_2 \sqrt{m}$ ，即  $\mathbf{v}$  的长度较短。

**原始攻击：变形1。** 当  $\alpha_1 = \alpha_2$  时，这种攻击算法即为目前最有效的原始攻击算法[8,6]。定义格

$$\Lambda = \{\mathbf{v} = (\mathbf{x}^T, \mathbf{y}^T, z)^T \in \mathbb{Z}^{n+m+1} | (\mathbf{A} \|\mathbf{I}_m\| - \mathbf{b})\mathbf{v} = 0 \bmod q\},$$

容易验证矩阵

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_n & 0 & 0 \\ -\mathbf{A} & q\mathbf{I}_m & \mathbf{b} \\ 0 & 0 & 1 \end{pmatrix} \in \mathbb{Z}^{(m+n+1) \times (m+n+1)}$$

的列向量构成格  $\Lambda$  的一组基。显然，格  $\Lambda$  维数为  $d = m + n + 1$ 。进一步，我们有  $\det(\Lambda) = q^m$ ，且  $\mathbf{v} = (\mathbf{s}^T, \mathbf{e}^T, 1)^T \in \mathbb{Z}^{n+m+1}$  是格  $\Lambda$  的一个短向量。因此，我们可以通过求解格  $\Lambda$  中的(u)SVP 问题来恢复向量  $\mathbf{s} \in \mathbb{Z}_q^m$ 。此外，由于向量  $\mathbf{s}$  和  $\mathbf{e}$  的每个分量都分别选自于亚高斯分布  $\chi_{\alpha_1}$  和  $\chi_{\alpha_2}$ ，我们有  $\|\mathbf{v}\| \approx \sqrt{\alpha_1^2 n + \alpha_2^2 m}$ 。

**原始攻击：变形2。** 这种攻击算法由文献[8,6]中原始攻击算法进一步变形而来。当  $\alpha_1 \neq \alpha_2$  且  $\alpha_1$  与  $\alpha_2$  的值相差不大时，该变形的原始攻击算法将更加有效。正式地，令  $c = \alpha_2/\alpha_1$ 。定义格

$$\Lambda = \left\{ \mathbf{v} = \begin{pmatrix} c\mathbf{x} \\ \mathbf{y} \\ \alpha_2 z \end{pmatrix} \in \mathbb{R}^{n+m+1} \mid (\mathbf{A} \|\mathbf{I}_m\| - \mathbf{b})\mathbf{u} = 0 \bmod q, \mathbf{u} = \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \\ z \end{pmatrix} \in \mathbb{Z}^{n+m+1} \right\},$$

则以下矩阵

$$\mathbf{B} = \begin{pmatrix} c\mathbf{I}_n & 0 & 0 \\ -\mathbf{A} & q\mathbf{I}_m & \mathbf{b} \\ 0 & 0 & \alpha_2 \end{pmatrix} \in \mathbb{R}^{(n+m+1) \times (n+m+1)},$$

的列向量构成格 $\Lambda$ 的一组基。显然，格 $\Lambda$ 维数为 $d = m + n + 1$ 。进一步，我们有 $\det(\Lambda) = \alpha_2 c^n q^m$ ，且 $\mathbf{v} = (c\mathbf{s}^T, \mathbf{e}^T, \alpha_2)^T \in \mathbb{R}^{m+n+1}$ 为格 $\Lambda$ 中的一个短向量。因此，我们可以通过求解格 $\Lambda$ 的(u)SVP问题来恢复私钥 $\mathbf{s}$ 。此外，由于向量 $\mathbf{s}$ 和 $\mathbf{e}$ 的每个分量都分别选自于亚高斯分布 $\chi_{\alpha_1}$ 和 $\chi_{\alpha_2}$ ，我们有 $\|\mathbf{v}\| \approx \alpha_2 \sqrt{n + m + 1}$ 。

直观上，该变形攻击算法的思想是将目标短向量的每个分量都放缩成同等的规模，从而达到更好的攻击效果。事实上，实际实验表明格上(u)SVP问题求解算法的确更加倾向于输出每个分量值都比较平衡的短向量。

**原始攻击的计算代价评估模型.** 目前求解最短向量最有效的办法为BKZ- $b$ 格基约化算法及其变形算法。给定 $d$ 维格的基作为输入，这类算法会将格基约化问题转变成 $b < d$ 维子格中的最短向量问题，并不断通过多次在不同 $b$ 维子格中求解最短向量问题来达到提高输入 $d$ 维格基质量的目的（即降低格基中向量的范数）。根据算法的不同，实际的BKZ- $b$ 算法通常会调用 $O(d)$ 次 $b$ 维格的最短向量求解算法。因此，BKZ- $b$ 格基约化算法的计算代价主要由 $b$ 维格的最短向量求解算法的计算代价来决定。

通常，我们假设运行BKZ- $b$ 算法对 $d$ 维格的基 $\mathbf{B} \in \mathbb{Z}^{d \times d}$ 进行约化得到的约化基满足几何级数假设，即GSA假设（该情况从攻击者角度为最优）。设得到的一组约化基 $\hat{\mathbf{B}} = (\hat{\mathbf{b}}_1, \dots, \hat{\mathbf{b}}_d)$ ，则有

$$\|\hat{\mathbf{b}}_1\| = \delta^d \det(\Lambda)^{1/d}, \quad \|\hat{\mathbf{b}}_i^*\| = \delta^{-2d(i-1)/(d-1)} \|\hat{\mathbf{b}}_1\|,$$

其中  $\delta = ((\pi b)^{1/b} \cdot b/2\pi e)^{\frac{1}{2(b-1)}}$  [16]， $\hat{\mathbf{B}}^* = (\hat{\mathbf{b}}_1^*, \dots, \hat{\mathbf{b}}_d^*)$  是矩阵 $\hat{\mathbf{B}}$ 的正交化矩阵（即 $\hat{\mathbf{b}}_1 = \hat{\mathbf{b}}_1^*$ ）。特别地，研究[6,4]表明当目标唯一最短向量 $\mathbf{v}$ 在最后 $b$ 个正交向量 $(\hat{\mathbf{b}}_{d-b+1}^*, \dots, \hat{\mathbf{b}}_d^*)$ 构成空间中的投影向量的范数小于 $\|\hat{\mathbf{b}}_{d-b+1}^*\|$ 时，那么运行BKZ- $b$ 格基约化算法将能够恢复向量 $\mathbf{v}$ 。

对于范数为 $\ell = \|\mathbf{v}\|$ 的唯一最短向量 $\mathbf{v}$ ，其在最后 $b$ 个正交向量 $(\hat{\mathbf{b}}_{d-b+1}^*, \dots, \hat{\mathbf{b}}_d^*)$ 构成空间中的投影向量的范数约为 $\ell \sqrt{b/d}$ ，即假设向量 $\mathbf{v}$ 在每个投影分量上的值近似相等。在这种情况下，原始攻击算法及其变形的计算代价主要由满足不等式

$$\ell \sqrt{b/d} \leq \delta^{(-d^2+2db-d)/(d-1)} \det(\Lambda)^{1/d} \tag{4}$$

的最小 $b$ 值来决定。与文献 [6] 中一样，我们将直接考虑用满足不等式(4)的 $b$ 维子格SVP求解算法的复杂度来非常保守地估计求解 $d$ 维格中唯一最短向量的复杂度

(因为后者需要将 $b$ 维子格SVP求解算法作为子算法运行非常多次)。进一步，与许多文献(例如[6,19,11])中一样，我们将分别使用 $\text{cost}_b = 2^{0.292b}$ 和 $\text{cost}_b = 2^{0.265b}$ 来分别刻画求解 $b$ 维格最短向量的经典算法和量子算法的复杂度。在这种保守模型下，原始攻击及其变形算法的复杂度主要由满足不等式(4)的 $b$ 值来确定。

## 6.2 对偶攻击及其变形

对偶攻击首先将LWE问题转化为SIS问题，然后利用SIS问题的解来将求解LWE问题转化成区分一定参数下的亚高斯分布和 $\mathbb{Z}_q^n$ 上的均匀分布的问题。我们将考虑以下针对ALWE问题 $\text{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ 的对偶攻击及其变形算法。

**传统对偶攻击.** 文献[31]首先给出了对偶攻击的方法。由于传统对偶攻击只考虑LWE问题的噪音分布，该类攻击方法实际上并不区分ALWE问题和LWE问题，即直接将ALWE问题看成LWE问题。特别地，当 $\alpha_1 \gg \alpha_2$ 时，传统对偶攻击比较有效。正式地，令

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T \mathbf{x} = \mathbf{0} \bmod q\},$$

则 $\Lambda$ 的维数 $d = m$ 。当 $m$ 足够大于 $n$ 时，矩阵 $\mathbf{A}$ 以很大概率存在 $n$ 个线性无关的列。不妨设 $\mathbf{A}^T$ 的前 $n$ 列线性无关(否则我们可以通过列变换将线性无关的列置换到前 $n$ 列)，且设前 $n$ 列对应的子矩阵为 $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ ，即 $\mathbf{A}^T = (\mathbf{A}_1 \parallel \mathbf{A}_2) \in \mathbb{Z}_q^{n \times m}$ 。记矩阵 $\mathbf{A}_1^{-1} \in \mathbb{Z}_q^{n \times n}$ 为 $\mathbf{A}_1$ 的逆，令 $\mathbf{A}' = (\mathbf{I}_n \parallel \mathbf{A}_1^{-1} \mathbf{A}_2)$ ，那么我们有

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T \mathbf{x} = \mathbf{0} \bmod q\} = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}' \mathbf{x} = \mathbf{0} \bmod q\},$$

且矩阵

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A}_1^{-1} \mathbf{A}_2 \\ 0 & \mathbf{I}_{m-n} \end{pmatrix} \in \mathbb{Z}^{m \times m}$$

的列向量构成格 $\Lambda$ 的一组基，显然 $\det(\Lambda) = q^n$ 。首先，传统对偶攻击将从以 $\mathbf{B}$ 为基的 $m$ 维格 $\Lambda$ 中寻找一个短向量 $\mathbf{v} \in \mathbb{Z}^m$ 满足 $\mathbf{A}^T \mathbf{v} = \mathbf{0} \bmod q$  (即求解SIS问题)。此后，攻击算法将计算 $u = \langle \mathbf{v}, \mathbf{b} \rangle = \langle \mathbf{v}, \mathbf{e} \rangle \bmod q$ ，并使用已知的算法来区分 $u$ 和选自于 $\mathbb{Z}_q$ 上均匀分布的元素。特别地，如果 $\ell = \|\mathbf{v}\|$ 比较小，那么 $\langle \mathbf{v}, \mathbf{e} \rangle$ 的值比较小，且元素 $u$ 可大致看作服从于以 $\ell\alpha_2$ 为标准差的亚高斯分布。在这种情况下，存在有效算法能够以 $4 \exp(-2\pi^2\tau^2)$ 的概率区分 $u$ 和 $\mathbb{Z}_q$ 中均匀分布的元素，其中 $\tau = \ell\alpha_2/q$ 。

显然，对于ALWE问题而言， $\alpha_2$ 取值越大， $\ell\alpha_2$ 的值也就越大，从而将 $u$ 和选自于 $\mathbb{Z}_q$ 中均匀分布的元素区分开来的概率就越小。因此，直观上， $\alpha_2$ 的值越大，传统对偶攻击算法将更难对ALWE进行攻击。

**对偶攻击：变形1.** 该变形的对偶攻击由[6]中的对偶攻击算法扩展而来。事实上，当 $\alpha_1 = \alpha_2$ ，该变形的算法就是[6]中的对偶攻击算法。正式地，定义格

$$\Lambda = \{(\mathbf{x}^T, \mathbf{y}^T)^T \in \mathbb{Z}^{m+n} | \mathbf{A}^T \mathbf{x} = \mathbf{y} \bmod q\},$$

则格 $\Lambda$ 的维数 $d = m + n$ 。当 $m$ 足够大于 $n$ 时，矩阵 $\mathbf{A}^T$ 以很大概率存在 $n$ 个线性无关的列。不妨设 $\mathbf{A}^T$ 的前 $n$ 列线性无关（否则我们可以通过列变换将线性无关的列置换到前 $n$ 列），且设前 $n$ 列对应的子矩阵为 $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ ，即  $\mathbf{A}^T = (\mathbf{A}_1 \| \mathbf{A}_2) \in \mathbb{Z}_q^{n \times m}$ 。记矩阵 $\mathbf{A}_1^{-1} \in \mathbb{Z}_q^{n \times n}$ 为 $\mathbf{A}_1$ 的逆，令 $\mathbf{A}' = (\mathbf{I}_n \| \mathbf{A}_1^{-1} \mathbf{A}_2)$ ，那么我们有

$$\Lambda = \{(\mathbf{x}^T, \mathbf{y}^T)^T | \mathbf{A}^T \mathbf{x} = \mathbf{y} \bmod q\} = \{(\mathbf{x}^T, \mathbf{y}^T)^T | \mathbf{A}' \mathbf{x} = \mathbf{A}_1^{-1} \mathbf{y} \bmod q\},$$

且矩阵

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A}_1^{-1} \mathbf{A}_2 & \mathbf{A}_1^{-1} \\ 0 & \mathbf{I}_{m-n} & 0 \\ 0 & 0 & \mathbf{I}_n \end{pmatrix} \in \mathbb{Z}^{(m+n) \times (m+n)}$$

的列向量构成格 $\Lambda$ 的一组基，显然 $\det(\Lambda) = q^n$ 。首先，该变形的对偶攻击将从以 $\mathbf{B}$ 为基的 $m + n$ 维格 $\Lambda$ 中寻找一个短向量 $\mathbf{v} = (\mathbf{x}^T, \mathbf{y}^T)^T \in \mathbb{Z}^{m+n}$ 满足 $\mathbf{A}^T \mathbf{x} = \mathbf{y} \bmod q$ 。此后，攻击算法将计算

$$u = \langle \mathbf{x}, \mathbf{b} \rangle = \langle \mathbf{y}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle \bmod q,$$

并使用已知的算法来区分 $u$ 和选自于 $\mathbb{Z}_q$ 上均匀分布的元素。特别地，如果 $\ell = \|\mathbf{v}\|$ 比较小，那么 $\langle \mathbf{y}, \mathbf{s} \rangle$ 与 $\langle \mathbf{x}, \mathbf{e} \rangle$ 的值也都比较小。进一步，若假设 $\mathbf{v}$ 中每个分量的值大致具有相同规模，那么元素 $u = \langle \mathbf{y}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle$ 可看作服从于以 $\ell \sqrt{\frac{\alpha_1^2 m + \alpha_2^2 n}{m+n}}$ 为标准差的亚高斯分布。在这种情况下，存在有效算法能够以 $4 \exp(-2\pi^2 \tau^2)$ 的概率区分 $u$ 和 $\mathbb{Z}_q$ 中均匀分布的元素，其中 $\tau = \ell \sqrt{\frac{\alpha_1^2 m + \alpha_2^2 n}{m+n}} / q$ 。

**对偶攻击：变形2.** 当 $\alpha_1 = \alpha_2$ ，对应ALWE问题退化为正规形LWE问题，且该变形的对偶攻击代表了目前针对正规形LWE问题的最有效的对偶攻击算法[6]。而 $\alpha_1 \neq \alpha_2$ 但 $\alpha_1$ 与 $\alpha_2$ 相差不多时，该变形的对偶攻击在求解ALWE问题时则更加有效。正式地，令 $c = \frac{\alpha_2}{\alpha_1}$ ，定义格

$$\Lambda = \{(\mathbf{x}^T, \mathbf{y}^T/c)^T \in \mathbb{R}^{m+n} | \mathbf{A}^T \mathbf{x} = \mathbf{y} \bmod q, \mathbf{x} \in \mathbb{Z}^m, \mathbf{y} \in \mathbb{Z}^n\},$$

则格 $\Lambda$ 的维数 $d = m + n$ 。当 $m$ 足够大于 $n$ 时，矩阵 $\mathbf{A}^T$ 以很大概率存在 $n$ 个线性无关的列。不妨设 $\mathbf{A}^T$ 的前 $n$ 列线性无关（否则我们可以通过列变换将线性无关的列置换到前 $n$ 列），且设前 $n$ 列对应的子矩阵为 $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ ，即  $\mathbf{A}^T = (\mathbf{A}_1 \| \mathbf{A}_2) \in \mathbb{Z}_q^{n \times m}$ 。记矩阵 $\mathbf{A}_1^{-1} \in \mathbb{Z}_q^{n \times n}$ 为 $\mathbf{A}_1$ 的逆，令 $\mathbf{A}' = (\mathbf{I}_n \| \mathbf{A}_1^{-1} \mathbf{A}_2)$ ，那么我们有

$$\Lambda = \{(\mathbf{x}^T, \mathbf{y}^T/c)^T | \mathbf{A}^T \mathbf{x} = \mathbf{y} \bmod q\} = \{(\mathbf{x}^T, \mathbf{y}^T/c)^T | \mathbf{A}' \mathbf{x} = \mathbf{A}_1^{-1} \mathbf{y} \bmod q\},$$

且矩阵

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A}_1^{-1}\mathbf{A}_2 & \mathbf{A}_1^{-1} \\ 0 & \mathbf{I}_{m-n} & 0 \\ 0 & 0 & \frac{1}{c}\mathbf{I}_n \end{pmatrix} \in \mathbb{R}^{(m+n) \times (m+n)}$$

的列向量构成格 $\Lambda$ 的一组基，显然有 $\det(\Lambda) = (q/c)^n$ 。首先，该变形的对偶攻击将从以 $\mathbf{B}$ 为基的 $m+n$ 维格 $\Lambda$ 中寻找一个短向量 $\mathbf{v} = (\mathbf{x}^T, \hat{\mathbf{y}}^T)^T \in \mathbb{R}^{m+n}$ 满足 $\mathbf{A}^T \mathbf{x} = c\hat{\mathbf{y}} \bmod q$ 。此后，攻击算法将计算

$$u = \langle \mathbf{x}, \mathbf{b} \rangle = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle \bmod q,$$

并使用已知的算法来区分 $u$ 和选自于 $\mathbb{Z}_q$ 上均匀分布的元素。特别地，如果 $\ell = \|\mathbf{v}\|$ 比较小，那么 $c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle$ 与 $\langle \mathbf{x}, \mathbf{e} \rangle$ 的值也都比较小，且元素 $u = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle$ 可大致看作服从于以 $\ell\alpha_2$ 为标准差的亚高斯分布。在这种情况下，存在有效算法能够以 $4 \exp(-2\pi^2\tau^2)$ 的概率区分 $u$ 和 $\mathbb{Z}_q$ 中均匀分布的元素，其中 $\tau = \ell\alpha_2/q$ 。

最后，需要指出的是该变形的对偶攻击并不关心 $\alpha_1$ 和 $\alpha_2$ 的大小关系。事实上，我们也可以用 $c' = \alpha_1/\alpha_2$ 来进行相应的对偶攻击，但这种对偶攻击在我们的计算代价评估模型下与用系数 $c = \alpha_2/\alpha_1$ 是等价的。

**对偶攻击：变形3.** 该变形的对偶攻击算法由文献[3]中的对偶攻击算法推广而来。该算法与变形2中的对偶攻击算法类似，其主要区别在于 $c$ 的取值不同。正式地，令 $c = \frac{\alpha_2\sqrt{m}}{\alpha_1\sqrt{n}}$ 。该变形的对偶攻击将首先计算一个短向量 $\mathbf{v} = (\mathbf{x}^T, \hat{\mathbf{y}}^T)^T \in \mathbb{R}^{m+n}$ 满足 $\mathbf{A}^T \mathbf{x} = c\hat{\mathbf{y}} \bmod q$ 。此后，攻击算法将计算

$$u = \langle \mathbf{x}, \mathbf{b} \rangle = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle \bmod q,$$

并使用已知的算法来区分 $u$ 和选自于 $\mathbb{Z}_q$ 上均匀分布的元素。特别地，如果 $\ell = \|\mathbf{v}\|$ 比较小，那么 $c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle$ 与 $\langle \mathbf{x}, \mathbf{e} \rangle$ 的值也都比较小，且元素 $u = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle$ 可大致看作服从于以 $\ell\alpha_2\sqrt{\frac{2m}{m+n}}$ 为标准差的亚高斯分布。此时，存在有效算法能够以 $4 \exp(-2\pi^2\tau^2)$ 的概率区分 $u$ 和 $\mathbb{Z}_q$ 中均匀分布的元素，其中 $\tau = \ell\alpha_2\sqrt{\frac{2m}{m+n}}/q$ 。

类似地，我们也可以用系数 $c' = \frac{\alpha_1\sqrt{n}}{\alpha_2\sqrt{m}}$ 来进行相应的对偶攻击，但这种情况在我们的计算代价评估模型下与用系数 $c = \frac{\alpha_2\sqrt{m}}{\alpha_1\sqrt{n}}$ 是等价的。

**对偶攻击的计算代价评估模型.** 如上所述，对偶攻击及其变形算法的计算代价主要由求解短向量问题的计算代价和将求ALWE问题的区分优势转变为计算优势的代价组成（其中后者要求区分优势至少应该大于 $1/2$ ）。对于求解短向量问题的代价，我们将使用原始攻击中的模型，即运行BKZ- $b$ 约化 $d$ 维格的基，我们将得到范数为 $\ell = \|\mathbf{v}\| = \delta^d \det(\Lambda)^{1/d}$ 的短向量 $\mathbf{v}$ 。将此代入此前的分析，我们将

得到区分ALWE问题的优势为 $\epsilon = 4 \exp(-2\pi^2\tau^2)$ , 其中 $\tau$ 由 $\ell$ 和具体的对偶攻击算法来唯一确定。为了使得最终的区分优势大于 $1/2$ , 我们将需要 $1/\epsilon^2$ 个短向量。考虑到每次运行筛法大约能够产生 $2^{0.2075b}$ 个短向量, 那么整个攻击需要重复大约 $R = 1 / \max(1, 1/(2^{0.2075b}\epsilon^2))$ 次。

如在原始攻击的计算代价模型一样, 我们将使用 $b$ 维子格SVP求解算法的复杂度来非常保守地估计求解 $d$ 维格中最短向量的复杂度 (因为后者需要将 $b$ 维子格SVP求解算法作为子算法运行非常多次)。进一步, 我们将使用 $\text{cost}_b = 2^{0.292b}$ 和 $\text{cost}_b = 2^{0.265b}$ 来分别刻画求解 $b$ 维格最短向量的经典算法和量子算法的复杂度。在这种保守模型下, 对偶攻击最终复杂度为

$$1 / \max(1, 1/(2^{0.2075b} \cdot 16 \exp(-4\pi^2\tau^2))) \cdot \text{cost}_b, \quad (5)$$

其中 $\tau$ 由 $b$ 和具体的对偶攻击算法来唯一确定。

### 6.3 埃奎斯密钥封装机制的安全强度

表 4. 埃奎斯密钥封装机制抵抗原始攻击的安全强度

参数集名称	攻击模型	传统原始攻击 ( $m, b, sec$ )	原始攻击变形1 ( $m, b, sec$ )	原始攻击变形2 ( $m, b, sec$ )
<b>PARAMS I</b>	经典	(763, 565, 165)	(441, 410, 119)	<b>(476, 385, 112)</b>
	量子	(763, 565, 149)	(441, 410, 108)	<b>(476, 385, 102)</b>
<b>PARAMS II</b>	经典	(1022, 830, 242)	(646, 575, 168)	<b>(556, 560, 163)</b>
	量子	(1022, 830, 220)	(646, 575, 152)	<b>(556, 560, 148)</b>
<b>PARAMS III</b>	经典	(1531, 1030, 301)	(886, 835, 244)	<b>(786, 815, 238)</b>
	量子	(1531, 1030, 273)	(886, 835, 221)	<b>(786, 815, 216)</b>

表 5. 埃奎斯密钥封装机制抵抗对偶攻击的安全性强度

参数集名称	攻击模型	传统对偶攻击 ( $m, b, sec$ )	对偶攻击变形1 ( $m, b, sec$ )	对偶攻击变形2 ( $m, b, sec$ )	对偶攻击变形 3 ( $m, b, sec$ )
<b>PARAMS I</b>	经典	(763, 560, 163)	(736, 395, 115)	(431, 385, 112)	<b>(521, 385, 112)</b>
	量子	(763, 560, 148)	(736, 395, 104)	<b>(431, 385, 102)</b>	<b>(521, 385, 102)</b>
<b>PARAMS II</b>	经典	(1022, 820, 239)	(881, 570, 166)	<b>(586, 555, 162)</b>	<b>(776, 555, 162)</b>
	量子	(1022, 820, 217)	(881, 570, 151)	<b>(586, 555, 147)</b>	<b>(776, 555, 147)</b>
<b>PARAMS III</b>	经典	(1531, 1020, 298)	(1446, 820, 239)	(906, 805, 236)	<b>(1171, 805, 235)</b>
	量子	(1531, 1020, 270)	(1446, 820, 217)	(906, 805, 214)	<b>(1171, 805, 213)</b>

由于攻击算法可能选择不同的样本数量和不同参数 $b \in \mathbb{Z}$ 来运行BKZ- $b$ 算法, 为了全面评估埃奎斯密钥封装机制在3组参数下的安全强度, 我们选择穷举所有

可能ALWE样本数量 $m$ 和格基约化算法BKZ- $b$ 的 $b$ 值来估计在不同原始攻击和对偶攻击下的最优计算复杂度，设其为 $2^{sec}$ 。特别地，表4给出了埃奎斯密钥封装机制三组参数在不同原始攻击下的实现最小攻击复杂度的 $(m, b, sec)$ 值。表5给出了埃奎斯密钥封装机制三组参数在不同对偶攻击下的实现最小攻击复杂度的 $(m, b, sec)$ 值。表6给出了埃奎斯密钥封装机制三组参数达到的整体安全强度。

**表 6.** 埃奎斯密钥封装机制的整体安全强度

参数集名称	经典安全性	量子安全性
<b>PARAMS I</b>	112	102
<b>PARAMS II</b>	162	147
<b>PARAMS III</b>	235	213

此外，为了研究非对称(M)LWE问题和标准(M)LWE问题的困难关系，针对埃奎斯密钥封装机制的3组参数集，在表7我们给出了3组MLWE问题的对比参数（但除了安全性外，对比参数并不能达到三组原始参数的正确性和效率要求）。实验结果显示A(M)LWE 问题 $\text{AMLWE}_{n,q,m,\alpha_1,\alpha_2}$ 问题和 $\text{MLWE}_{n,q,m,\sqrt{\alpha_1\alpha_2}}$ 问题在我们的模型下的复杂度近似相同，即

$$\text{AMLWE}_{n,q,m,\alpha_1,\alpha_2} \approx \text{MLWE}_{n,q,m,\sqrt{\alpha_1\alpha_2}}$$

**表 7.** 3组MLWE问题对比参数集的安全强度

参数集名称	$(n, k, q, \eta_1, \eta_2)$	经典安全强度	量子安全强度
<b>PARAMS I</b>	(256, 2, 7681, 2, 12)	112	102
<b>MLWE对比参数I</b>	(256, 2, 7681, 5, 5)	112	102
<b>PARAMS II</b>	(256, 3, 7681, 1, 4)	162	147
<b>MLWE对比参数II</b>	(256, 3, 7681, 2, 2)	163	148
<b>PARAMS III</b>	(512, 2, 12289, 2, 8)	235	213
<b>MLWE对比参数III</b>	(512, 2, 12289, 4, 4)	236	214

## 7 优缺点

总的来说，我们基于一个变种的(M)LWE困难问题（即非对称(M)LWE问题）构造了一类高效的、选择密文安全的密钥封装机制。在理论复杂性上，该类变种问题与标准(M)LWE问题在渐进意义下是等价的。在实际安全强度上，我们分析了已有(M)LWE问题的最优攻击算法，提出了针对非对称(M)LWE问题的改进

算法，给出了具体安全参数的评估方法。具体分析和实验结果显示，利用非对称(M)LWE困难问题设计的埃奎斯密钥封装机制能够实现更好的综合效率，达到更短的公钥和密文长度，提供更加灵活的参数选取。

**优点.** 我们提出的埃奎斯密钥封装机制具有以下优点：

- **安全性高:** 埃奎斯密钥封装机制在经典随机预言机模型和量子随机预言机模型下都是可证明选择密文攻击安全的。此外，在参数选取和安全评估方面，我们使用了非常保守的安全评估模型（见第6节），从而能够在一定程度上抵抗未来改进的经典和量子攻击方法。
- **公钥和密文长度短:** 与格上同类方案比较，埃奎斯密钥封装机制具有更短的公钥和密文长度。例如，对于保守评估安全强度达到147比特量子安全强度，解密错误率小于 $2^{-128}$ 的参数集，其对应公钥和密文的长度也分别只有896字节和992字节。
- **速度快:** 埃奎斯密钥封装机制具有非常高效的密钥生成、密钥封装和解封装算法。例如，对于保守评估安全强度达到213比特量子安全强度、解密错误率小于 $2^{-211}$ 、封装512比特（64字节）密钥的参数集，在安装Windows 10操作系统的普通笔记本（2.5 GHz CPU）上，标准C语言实现的密钥生成、密钥封装和解封装算法的运行时间分别只需大概0.12ms、0.18ms和0.18ms，而用AVX2指令部分优化实现的算法则进一步分别降低到0.05ms、0.07ms和0.06ms。
- **良好的灵活性:**
  - **用途灵活:** 通过将埃奎斯密钥封装机制与（一次安全的）对称加密方案组合就能获得加密任意明文长度的选择密文攻击安全的公钥加密方案[18]。此外，根据通用转换方法[11,20]，我们还能基于埃奎斯密钥封装机制构造高效的两轮密钥交换协议和两轮认证密钥交换协议。
  - **参数选取灵活:** 与基于标准(M)LWE困难问题的密钥封装机制比较（例如[6,11]），埃奎斯密钥封装机制支持更加灵活细粒度的参数选取，从而更容易实现安全和性能的平衡。
- **抵抗多目标攻击:** 埃奎斯密钥封装机制采用了多种方式抵抗多目标攻击。特别地，不同用户的公钥采用了不同的随机矩阵A，从而阻止了攻击者以恢复一个用户私钥的代价来恢复多个用户的私钥。同时，埃奎斯密钥封装机制通过将用户公钥与封装密钥和密文中的随机数相绑定，使得攻击者不能用一次预算的结果来攻击多个用户。
- **易于安全实现:** 埃奎斯密钥封装机制没有使用高斯分布，且没有通过使用高级的纠错码来降低方案的错误率，从而能避免相关针对高斯分布采样算法和纠错码译码算法实现的侧信道攻击。

**缺点.** 与许多格上高效的密码系统一样，埃奎斯密钥封装机制采用了环结构。虽然我们使用的 AMLWE 问题有望能够提供比普通环上 LWE 问题更强的安全保证，但与标准 LWE 问题相比，环结构的使用仍有可能留给量子敌手更多攻击的空间。

## 参考文献

1. Damien Stehlé Adeline Langlois. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.
2. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, STOC ’96, pages 99–108, New York, NY, USA, 1996. ACM.
3. Martin R. Albrecht. On dual lattice attacks against small-secret lwe and parameter choices in helib and seal. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, pages 103–129, Cham, 2017. Springer International Publishing.
4. Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the expected cost of solving usvp and applications to lwe. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 297–322, Cham, 2017. Springer International Publishing.
5. Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. In *Journal of Mathematical Cryptology*, 9:169–203, oct 2015.
6. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange—a new hope. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 327–343, Austin, TX, 2016. USENIX Association.
7. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer Berlin Heidelberg, 2009.
8. Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary lwe. In Willy Susilo and Yi Mu, editors, *Information Security and Privacy*, pages 322–337, Cham, 2014. Springer International Publishing.
9. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy Denning, Ray Pyle, Ravi Ganesan, Ravi Sandhu, and Victoria Ashby, editors, *First ACM Conference on Computer and Communication Security*, pages 62–73. ACM, November 3–5 1993.
10. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In DongHoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69. Springer Berlin Heidelberg, 2011.
11. J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehle. Crystals - kyber: A cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 353–367, April 2018.
12. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. Fully homomorphic encryption without bootstrapping. *Innovations in Theoretical Computer Science, ITCS*, pages 309–325, 2012.
13. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 97–106, oct. 2011.

14. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 575–584, New York, NY, USA, 2013. ACM.
15. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 505–524. Springer Berlin Heidelberg, 2011.
16. Yuanmi Chen. Lattice reduction and concrete security of fully homomorphic encryption. *Dept. Informatique, ENS, Paris, PhD thesis*, 2013.
17. Jung Hee Cheon, Duhyeong Kim, Joohee Lee, and Yongsoo Song. Lizard: Cut off the tail! a practical post-quantum public-key encryption from lwe and lwr. In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks*, pages 160–177, Cham, 2018. Springer International Publishing.
18. Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33:167–226, 2001.
19. Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018.
20. Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography – PKC 2012*, volume 7293 of *Lecture Notes in Computer Science*, pages 467–484. Springer Berlin Heidelberg, 2012.
21. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO 1999*, volume 1666 of *LNCS*, pages 537–554. Springer, 1999.
22. Shafi Goldwasser, Yael Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In *Proceedings of the Innovations in Computer Science 2010*. Tsinghua University Press, 2010.
23. S.Dov Gordon, Jonathan Katz, and Vinod Vaikuntanathan. A group signature scheme from lattice assumptions. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 395–412. Springer Berlin / Heidelberg, 2010.
24. Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the fujisaki-okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography – TCC 2017*, volume 10677 of *LNCS*, pages 341–371. Springer International Publishing, 2017.
25. Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, volume 10993 of *LNCS*, pages 96–125. Springer International Publishing, 2018.
26. Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.
27. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *Topics in Cryptology – CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer Berlin Heidelberg, 2011.
28. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer Berlin / Heidelberg, 2010.

29. Daniele Micciancio. On the hardness of learning with errors with binary secrets. *Theory of Computing*, 14(13):1–17, 2018.
30. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 372 – 381, 2004.
31. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In DanielJ. Bernstein, Johannes Buchmann, and Erik Dahmen, editors, *Post-Quantum Cryptography*, pages 147–191. Springer Berlin Heidelberg, 2009.
32. National Institute of Standards and Technology. Sha-3 standard: Permutation-based hash and extendable-output functions. FIPS PUB 202, 2015. <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf>.
33. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC ’09, pages 333–342, New York, NY, USA, 2009. ACM.
34. Chris Peikert. An efficient and parallel gaussian sampler for lattices. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97. Springer Berlin Heidelberg, 2010.
35. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC ’05, pages 84–93, New York, NY, USA, 2005. ACM.

## A 安全模型

### A.1 公钥加密方案定义

**公钥加密方案的语法.** 消息空间为 $\mathcal{M}$ 的公钥加密方案 $\text{PKE} = (\text{KeyGen}, \text{Enc}, \text{Dec})$ 由以下算法组成:

**密钥生成算法:** 概率算法 $\text{KeyGen}$ 输出一对公私钥 $(pk, sk)$ 。密钥生成过程表示为 $(pk, sk) \leftarrow \text{KeyGen}(\kappa)$ 。

**加密算法:** 概率算法 $\text{Enc}$ 输入公钥 $pk$ 和消息(也叫明文) $\mu \in \mathcal{M}$ , 输出密文 $c$ 。加密过程表示为 $c \leftarrow \text{Enc}(pk, \mu)$ 。

**解密算法:** 确定性算法 $\text{Dec}$ 输入私钥 $sk$ 和密文 $c$ , 输出消息 $\mu \in \mathcal{M}$ 或一个特殊符号 $\perp$ 去代表拒绝。解密过程表示为 $\mu / \perp \leftarrow \text{Dec}(sk, c)$ 。

**正确性.** 我们称公钥加密方案 $\text{PKE}$ 是 $(1 - \delta)$ -正确的, 如果对于所有消息 $\mu \in \mathcal{M}$ 以下成立:

$$\Pr [ (pk, sk) \leftarrow \text{KeyGen}(\kappa) : \text{Dec}(sk, \text{Enc}(pk, \mu)) = \mu ] \geq 1 - \delta,$$

其中概率取自 $\text{KeyGen}$ 和 $\text{Enc}$ 随机投币之上。

**定义 3 (IND-CPA)** 我们称公钥加密方案PKE满足在选择明文攻击下不可区分性(即IND-CPA安全性), 如果以下定义的敌手优势 $\text{Adv}_{\text{PKE}}^{\text{ind}-\text{cpa}}(\mathcal{A})$ 是可忽略的,

$$\text{Adv}_{\text{PKE}}^{\text{ind}-\text{cpa}}(\mathcal{A}) := \left| \Pr \left[ b = b' \middle| \begin{array}{l} (pk, sk) \leftarrow \text{KeyGen}(); \\ (\mu_0, \mu_1, st) \leftarrow \mathcal{A}(pk); \\ b \xleftarrow{\$} \{0, 1\}; c^* \leftarrow \text{Enc}(pk, \mu_b); \\ b' \leftarrow \mathcal{A}(st, c^*) \end{array} \right] - \frac{1}{2} \right|,$$

其中敌手输出的明文要求长度相同, 即 $|\mu_0| = |\mu_1|$ 。

## A.2 密钥封装机制定义

**密钥封装机制语法.** 密钥空间为 $\mathcal{K}$ 的密钥封装机制 $\text{KEM} = (\text{KeyGen}, \text{Encaps}, \text{Decaps})$ 由以下算法组成:

**密钥生成算法:** 概率算法 $\text{KeyGen}$ 输出一对公私钥 $(pk, sk)$ 。密钥生成过程表示为 $(pk, sk) \leftarrow \text{KeyGen}(\kappa)$ 。

**密钥封装算法:** 概率算法 $\text{Encaps}$ 输入公钥 $pk$ , 输出一个密文 $c$ 和密钥 $K \in \mathcal{K}$ 。封装过程表示为 $(c, K) \leftarrow \text{Encaps}(pk)$ 。

**解封装算法:** 确定性算法 $\text{Decaps}$ 输入私钥 $sk$ 和密文 $c$ , 输出密钥 $K \in \mathcal{K}$ 或一个特殊符号 $\perp$ 去代表拒绝。解封过程表示为 $K / \perp \leftarrow \text{Decaps}(sk, c)$ 。

**正确性.** 我们称密钥封装机制 $\text{KEM}$ 是 $(1 - \delta)$ -正确的, 如果以下成立:

$$\Pr [ (pk, sk) \leftarrow \text{KeyGen}(\kappa), (c, K) \leftarrow \text{Encaps}(pk) : \text{Decaps}(sk, c) = K ] \geq 1 - \delta,$$

其中概率取自 $\text{KeyGen}$ 和 $\text{Enc}$ 随机投币之上。

**定义 4 (IND-CCA2)** 我们称密钥封装机制 $\text{KEM}$ 满足在选择密文攻击下不可区分性(即IND-CCA2安全性), 如果以下定义的敌手优势 $\text{Adv}_{\text{KEM}}^{\text{ind}-\text{cca}}(\mathcal{A})$ 是可忽略的,

$$\text{Adv}_{\text{KEM}}^{\text{ind}-\text{cca}}(\mathcal{A}) := \left| \Pr \left[ b = b' \middle| \begin{array}{l} (pk, sk) \leftarrow \text{KeyGen}(\kappa); b \xleftarrow{\$} \{0, 1\}; \\ (c^*, K_0^*) \leftarrow \text{Encaps}(pk); K_1^* \xleftarrow{\$} \mathcal{K}; \\ b' \leftarrow \mathcal{A}^{\text{Decaps}(sk, \cdot)}(pk, c^*, K_b^*) \end{array} \right] - \frac{1}{2} \right|,$$

其中敌手 $\mathcal{A}$ 不允许用挑战密文 $c^*$ 询问解封装预言机 $\text{Decaps}(sk, \cdot)$ 。