中国科学: 信息科学 2021 年 第 51 卷 第 12 期: 2119–2133
SCIENTIA SINICA Informationis
论文

抗泄露的无证书密钥封装机制及应用

周彦伟[1,2,4], 杨波[1*], 乔子芮[1*], 夏喆[3], 张明武[2,4]

1. 陕西师范大学计算机科学学院, 西安 710062
2. 桂林电子科技大学广西密码学与信息安全重点实验室, 桂林 541004
3. 武汉理工大学计算机科学与技术学院, 武汉 430070
4. 密码科学技术国家重点实验室, 北京 100878

* 通信作者. E-mail: byang@snnu.edu.cn, qzr_snnu@163.com

收稿日期: 2021–01–01; 修回日期: 2021–02–18; 接受日期: 2021–02–24; 网络出版日期: 2021–11–22

国家重点研发计划 (批准号: 2017YFB0802000)、国家自然科学基金 (批准号: U2001205, 61802242, 61772326, 61802241)、广西密码学与信息安全重点实验室研究课题 (批准号: GCIS202108) 和中央高校基本科研业务费 (批准号: GK202003079, GK202007033) 资助项目

摘要 泄露攻击的出现, 导致在传统理想安全模型下已证明安全的密码机制在实际应用中不再保持其所声称的安全性; 并且现有基于双线性映射构造的抗泄露无证书密钥封装机制 (certificateless key-encapsulation mechanism, CL-KEM) 的计算效率较低. 针对上述不足, 在不使用双线性映射的前提下本文设计了抗连续泄露的 CL-KEM, 并基于经典的判定性 Diffie-Hellman 假设对构造的安全性进行形式化证明. 在我们的 CL-KEM 实例中, 封装密文的所有元素对敌手而言是随机的, 确保任意敌手均无法从封装密文中获知关于用户私钥的泄露信息; 并且泄露参数是固定的常数, 不受封装密钥空间大小的限制. 为了进一步增强 CL-KEM 的抗泄露攻击的能力, 本文构造了一个泄露量达到 $l_{\mathrm{sk}}(1-\mathcal{O}(1))$ 的新颖抗泄露 CL-KEM, 其中 l_{sk} 表示私钥的长度, 分析表明该机制在具有上述优势的同时, 将抵抗泄露攻击的能力提升到最佳. 最后, 基于抗泄露 CL-KEM 提出抗泄露无证书混合加密机制和抗泄露无证书密钥协商协议的通用构造方法.

关键词 无证书公钥密码, 密钥封装机制, 泄露容忍, 连续泄露容忍, DDH 安全性假设

1 引言

现有密码机制的安全性均是在理想模型下进行证明的, 该安全模型中敌手仅能获知密码机制的输入和输出, 无法接触内部秘密状态 (如随机数、私钥等). 遗憾的是, 现实应用中泄露攻击 (如边信道、冷启动等) 的存在, 使得敌手能够获得内部秘密状态的部分泄露, 导致传统密码机制在现实应用中不再具备相应的安全性. 为获得更加实用的密码机制, 需为其提供抵抗泄露攻击的能力. 此外通过研究密码体制的连续泄露容忍性达到抵抗连续泄露攻击的目的, 能够获得更接近现实应用环境的密码机制, 缩短密码理论研究与现实安全需求间的距离.

引用格式: 周彦伟, 杨波, 乔子芮, 等. 抗泄露的无证书密钥封装机制及应用. 中国科学: 信息科学, 2021, 51: 2119–2133, doi: 10.1360/SSI-2021-0001
Zhou Y W, Yang B, Qiao Z R, et al. Leakage-resilient certificateless key-encapsulation mechanism and application (in Chinese). Sci Sin Inform, 2021, 51: 2119–2133, doi: 10.1360/SSI-2021-0001

1.1 无证书密钥封装机制抗泄露性的研究现状

2003 年, Al-Riyami 等 [1] 提出无证书公钥密码体制 (certificateless public-key cryptography, CL-PKC) 减少了对可信第三方密钥生成中心 (key generation center, KGC) 的依赖. 在 CL-PKC 中, 用户基于 KGC 为其计算的部分私钥和随机选取的秘密值生成用户的完整私钥; 公钥由用户的秘密值、身份信息和系统参数计算得出, 并对外安全公布. CL-PKC 解决了基于身份密码体制 (identity-based cryptography, IBC) 中的用户密钥托管问题, 也消除了传统公钥基础设施中证书的复杂性管理问题, 提高了密码系统的运行效率, 上述优势使得 CL-PKC 在现实应用环境中具有广阔的应用前景. 近年来, 无证书密码原语的研究得到了研究者的广泛关注, 无证书密钥协商协议 [2]、无证书聚合签名机制 [3]、无证书公钥加密 [4] (certificateless public-key encryption, CL-PKE) 机制、无证书密钥封装机制 [5] (certificateless key-encapsulation mechanism, CL-KEM) 等原语相继被提出. 为提升相应密码机制抗泄露攻击的能力, 文献 [6,7] 设计了抗泄露的 CL-PKE 机制, 文献 [8] 提出了抗连续泄露的 CL-PKE 机制. 文献 [9] 提出了第 1 个抗泄露的 CL-KEM, 并在通用的双线性群模型中对机制的安全性进行了证明; 然而该机制仅讨论了有界泄露容忍性, 且双线性映射的使用导致其计算效率较低.

综上所述, 当前对抗泄露 CL-KEM 的研究成果较少, 并且仅有的研究 [9] 尚未达到最佳的计算效率和泄露容忍性. 为获得更加高效的抗泄露 CL-KEM, 本文将设计不使用双线性映射的抗泄露 CL-KEM; 同时为进一步获得最佳的泄露容忍性, 本文将设计抵抗连续泄露攻击的 CL-KEM; 并通过增加用户私钥最小熵的方法提升 CL-KEM 抗泄露攻击的能力. 此外, 由于 CL-KEM 能够跟数据封装机制配合组成无证书混合加密的形式, 无证书混合加密继承了无证书公钥加密的密钥管理优势, 还兼顾了对称密码的高计算效率优点; 同时, CL-KEM 能作为底层工具设计密钥协商协议. 因此本文对抗泄露 CL-KEM 的研究能进一步提升无证书混合加密机制和无证书密钥协商协议等密码原语的计算效率和泄露容忍性.

1.2 我们的工作

相较于现有的抗泄露 CL-KEM 而言 [9], 我们的构造具有更高的计算效率和更优的泄露容忍性. 本文的贡献主要分为 3 个方面: (1) 提出一个不使用双线性映射的抗连续泄露 CL-KEM, 且封装密文的所有元素对于任意敌手而言是随机的, 确保任何敌手都无法从封装密文中获得相关私钥的泄露信息; 此外保持泄露参数是固定常数, 与封装密钥空间的大小无关. 基于判定性 Diffie-Hellman (decisional Diffie-Hellman, DDH) 假设证明了方案选择密文攻击 (chosen-ciphertext attack, CCA) 下的不可区分性. (2) 为获得最佳的泄露容忍性, 设计泄露率为 $1-\mathcal{O}(1)$ 的新型抗泄露 CL-KEM. (3) 基于抗泄露 CL-KEM 提出抗泄露无证书混合加密机制和抗泄露无证书密钥协商协议的通用构造, 在实际应用层面推广了抗泄露 CL-KEM.

本文工作组织如下: 第 2 节介绍了强随机性提取器、困难性假设等相关的基础知识; 第 3 节介绍了无证书密钥封装机制的形式化定义及抗泄露的安全模型; 第 4 节设计了基础的抗泄露 CL-KEM, 该构造的密文元素对任意敌手而言是随机的, 且具有固定的泄露参数; 为实现抵抗连续泄露攻击的目的; 第 5 节通过对基础 CL-KEM 增加密钥更新算法实现私钥的周期性更新; 第 6 节在保持基础 CL-KEM 公开参数不变的前提下, 通过增加私钥长度获得最佳的泄露容忍性.

2 基础知识

本文中安全参数用 κ 表示; $\mathsf{negl}(\kappa)$ 表示在 κ 上计算可忽略的值; $a \leftarrow_R A$ 表示均匀随机的从集合 A 中选取一个元素 a.

2.1 强随机性提取器

令 $\mathsf{SD}(X,Y) = \frac{1}{2}\sum_{\omega\in\Omega}|\Pr[X=\omega]-\Pr[Y=\omega]|$ 表示任意两个随机变量 $X \in \Omega$ 与 $Y \in \Omega$ 间的统计距离, 其中 Ω 表示有限域; $H_\infty(X) = -\log(\max_x \Pr[X=x])$ 表示任意随机变量 X 的最小熵; $\widetilde{H}_\infty(X|Y) = \log(\mathrm{E}_z \max_x \Pr[X=x|Y=y]) = -\log(\mathrm{E}_{y\leftarrow Y}[2^{-H_\infty(X|Y=y)}])$ 表示已知变量 Y 时变量 X 的平均最小熵, 其中 E 表示数学期望运算 [10~12].

定理1 已知随机变量 Z 的取值最多有 2^k 个, 那么对于任意的随机变量 X 和 Y, 有 $\widetilde{H}_\infty(X|(Y,Z)) \geqslant \widetilde{H}_\infty(X|Y) - k$ 成立 [13].

定义1 (强随机性提取器) 对于满足条件 $X \in \{0,1\}^{l_n}$ 和 $\widetilde{H}_\infty(X|Y) \geqslant k$ 的任意随机变量 X 和 Y, 若有 $\mathsf{SD}((\mathsf{Ext}(X,S),S,Y),(Z,S,Y)) \leqslant \mathsf{negl}(\kappa)$ 成立, 其中 $S \leftarrow_R \{0,1\}^{l_t}$ 和 $Z \leftarrow_R \{0,1\}^{l_m}$, 则称 $\mathsf{Ext} : \{0,1\}^{l_n} \times \{0,1\}^{l_t} \to \{0,1\}^{l_m}$ 是平均情况的 (k,ε) – 强随机性提取器 [14].

2.2 困难性假设

令群生成算法 $\mathcal{G}(1^\kappa)$ 的输出是 (q,P,G), 其中 G 是阶为大素数 q 的循环群, P 是群 G 的生成元.

定义2 (DDH 假设) 已知 $(q,P,G) \leftarrow \mathcal{G}(1^\kappa)$ 和 $a,b,d \leftarrow_R Z_q^*$, 对于任意的概率多项式时间算法 $\mathcal{A}$, 其区分元组 (P,aP,bP,abP) 和 (P,aP,bP,dP) 的优势

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DBDH}}(\kappa) = |\Pr[\mathcal{A}(P,aP,bP,abP)=1] - \Pr[\mathcal{A}(P,aP,bP,dP)=1]|$$

是可忽略的, 其中概率来源于 a, b, d 的随机选取和算法 $\mathcal{A}$ 的随机选择.

2.3 密钥衍射函数

对于任意的敌手 $\mathcal{A}$, 有 $\mathsf{Adv}^{\mathrm{KDF}}(\kappa) \leqslant \mathsf{negl}(\kappa)$ 成立, 那么称 $\mathsf{KDF} : \{0,1\}^* \to \{0,1\}^{l_k}$ 是安全的密钥衍射函数, 其中 $\mathsf{Adv}^{\mathrm{KDF}}(\kappa) = |\Pr[b=b'] - \frac{1}{2}|$ 表示敌手 $\mathcal{A}$ 在下述游戏 $\mathbf{Game}_{\mathcal{A},\mathcal{C}}^{\mathsf{KDF}}(\kappa)$ 中获胜的优势 [15].

$\mathbf{Game}_{\mathcal{A},\mathcal{C}}^{\mathsf{KDF}}(\kappa)$:

(1) 挑战者 $\mathcal{C}$ 初始化系统环境, 并发送相应的公开参数给敌手 $\mathcal{A}$.

(2) $\mathcal{A}$ 选择随机的 $x \leftarrow \{0,1\}^*$ 发送给 $\mathcal{C}$.

(3) $\mathcal{C}$ 计算 $k_1 = \mathsf{KDF}(x)$, 并随机选取 $k_0 \leftarrow_R \{0,1\}^{l_k}$ 和 $\beta \leftarrow_R \{0,1\}$, 最后发送 k_β 给 $\mathcal{A}$.

(4) $\mathcal{A}$ 输出对 β 的猜测 β', 若 $\beta = \beta'$, 则 $\mathcal{A}$ 在该游戏中获胜.

3 无证书密钥封装机制的定义及抗泄露安全模型

本节将给出抗泄露 CL-KEM 的形式化定义及安全模型的详细介绍.

3.1 形式化定义

CL-KEM 包含下述 7 个算法, 具体描述如下:

(1) 初始化算法 $(\mathrm{Params}, \mathrm{msk}) \leftarrow \mathsf{Setup}(1^\kappa)$ 由 KGC 运行, 以 κ 为输入, 输出公开参数 Params 和主密钥 msk, 其中 Params 定义了身份空间 $\mathcal{ID}$, 封装密钥空间 $\mathcal{K}$, 封装密文空间 $\mathcal{C}$ 和私钥空间 $\mathcal{SK}$.

(2) 部分密钥生成算法 $y_{\mathrm{id}} \leftarrow \mathsf{Partial\text{-}Key\text{-}Extraction}(\mathrm{msk}, \mathrm{id})$ 由 KGC 运行, 输入身份 id 和 msk, 输出相应的部分私钥 y_{id}. 特别地, 多数情况下该算法在输出 y_{id} 的同时, 也会生成部分公钥 Y_{id}, 即 $(y_{\mathrm{id}}, Y_{\mathrm{id}}) \leftarrow \mathsf{Partial\text{-}key\text{-}Extraction}(\mathrm{msk}, \mathrm{id})$.

(3) 秘密值生成算法 $x_{\mathrm{id}} \leftarrow \mathsf{Set\text{-}Secret\text{-}Value}(\mathrm{id})$ 由用户执行, 输入身份 id, 输出相应的秘密值 x_{id}.

(4) 私钥生成算法 $\mathrm{sk}_{\mathrm{id}} \leftarrow \mathsf{Set\text{-}Private\text{-}Key}(x_{\mathrm{id}}, y_{\mathrm{id}})$ 由用户执行, 输入 KGC 生成的部分私钥 y_{id} 及秘密值 x_{id}, 输出相应的私钥 $\mathrm{sk}_{\mathrm{id}}$.

(5) 公钥生成算法 $\mathrm{pk}_{\mathrm{id}} \leftarrow \mathsf{Set\text{-}Public\text{-}Key}(x_{\mathrm{id}}, Y_{\mathrm{id}})$ 由用户执行, 输入秘密值 x_{id} (和 KGC 生成的部分公钥 Y_{id}), 输出相应的公钥 $\mathrm{pk}_{\mathrm{id}}$.

(6) 密钥封装算法 $(C, k) \leftarrow \mathsf{Encap}(\mathrm{id}, \mathrm{pk}_{\mathrm{id}})$ 由发送者负责执行. 输入接收者的身份 $\mathrm{id} \in \mathcal{ID}$ 和公钥 $\mathrm{pk}_{\mathrm{id}}$, 输出封装密文 C 及对应的封装密钥 k.

(7) 解封装算法 $k \leftarrow \mathsf{Decap}(C, \mathrm{sk}_{\mathrm{id}})$ 由接收者负责执行. 输入封装密文 C 和自己的私钥 $\mathrm{sk}_{\mathrm{id}}$, 输出相应的封装密钥 k.

特别地, 本文方案设计时, 使用一个交互式的密钥生成算法实现部分密钥生成、秘密值生成、私钥生成和公钥生成等 4 个算法的功能, 因此可将 CL-KEM 简写成由 Setup, KeyGen, Encap 和 Decap 4 个算法组成, 其中密钥生成算法 $(\mathrm{pk}_{\mathrm{id}}, \mathrm{sk}_{\mathrm{id}}) \leftarrow \mathsf{KeyGen}(\mathrm{msk}, \mathrm{id})$ 是用户与 KGC 间的交互式算法, 生成用户 id 的公钥 $\mathrm{pk}_{\mathrm{id}} = (X_{\mathrm{id}}, Y_{\mathrm{id}})$ 和私钥 $\mathrm{sk}_{\mathrm{id}} = (x_{\mathrm{id}}, y_{\mathrm{id}})$, 其中 x_{id} 是用户设定的秘密值, $(y_{\mathrm{id}}, Y_{\mathrm{id}})$ 是由 KGC 生成的部分密钥, X_{id} 是由秘密值 x_{id} 所决定的公开信息.

已有研究 [16] 表明, 在保持公开参数和功能不变的前提下, 通过定期执行密钥更新操作能够在抗有界泄露攻击密码机制的基础上获得相应的连续泄露容忍性. 密钥更新算法 $\mathrm{sk}'_{\mathrm{id}} \leftarrow \mathsf{Update}(\mathrm{sk}_{\mathrm{id}})$ 以用户的原始私钥 $\mathrm{sk}_{\mathrm{id}}$ 作为输入, 输出更新后的有效私钥 $\mathrm{sk}'_{\mathrm{id}}$, 并且有 $|\mathrm{sk}_{\mathrm{id}}| = |\mathrm{sk}'_{\mathrm{id}}|$ 和 $\mathsf{SD}(\mathrm{sk}'_{\mathrm{id}}, \mathrm{sk}_{\mathrm{id}}) \leqslant \mathsf{negl}(\kappa)$ 成立.

3.2 正确性

对于 $(\mathrm{Params}, \mathrm{msk}) \leftarrow \mathsf{Setup}(1^{\kappa})$, $(\mathrm{pk}_{\mathrm{id}}, \mathrm{sk}_{\mathrm{id}}) \leftarrow \mathsf{KeyGen}(\mathrm{msk}, \mathrm{id})$ 和 $\mathrm{sk}'_{\mathrm{id}} \leftarrow \mathsf{Update}(\mathrm{sk}_{\mathrm{id}})$, 下述关系成立:

$$\Pr[k \neq k' | (C, k) \leftarrow \mathsf{Encap}(\mathrm{id}, \mathrm{pk}_{\mathrm{id}}), k' \leftarrow \mathsf{Decap}(C, \mathrm{sk}_{\mathrm{id}})] \leqslant \mathsf{negl}(\kappa);$$
$$\Pr[k \neq k'' | (C, k) \leftarrow \mathsf{Encap}(\mathrm{id}, \mathrm{pk}_{\mathrm{id}}), k'' \leftarrow \mathsf{Decap}(C, \mathrm{sk}'_{\mathrm{id}})] \leqslant \mathsf{negl}(\kappa).$$

3.3 容忍泄露的安全模型

由于 KGC 仅生成了用户的部分密钥, 因此在无证书密码体制中不再假设 KGC 是完全可信的第三方. 由于现实应用中没有绝对可信的第三方, 使得该机制具有广泛的应用前景. 对于无证书密码机制而言, 恶意的用户将通过替换他人公钥的方式对该技术展开攻击; 而恶意的 KGC 将利用其掌握主私钥的优势对其进行攻击. 因此, CL-KEM 将受到两类敌手 $\mathcal{A}^1$ 和 $\mathcal{A}^2$ 的攻击.

(1) 敌手 $\mathcal{A}^1$ 具有替换合法用户公钥的能力, 但无法掌握主私钥, 则 $\mathcal{A}^1$ 为恶意的用户. 但是 $\mathcal{A}^1$ 不能对挑战身份进行私钥和部分密钥生成询问, 并且在挑战阶段之前不能替换挑战身份的公钥.

(2) 敌手 $\mathcal{A}^2$ 拥有主私钥, 但不能替换合法用户的公钥, 则 $\mathcal{A}^2$ 为恶意的 KGC. 但是 $\mathcal{A}^2$ 不能对挑战身份进行私钥生成询问, 并且不能替换任何用户的公钥.

在抗泄露的安全模型中, 敌手通过访问泄露谕言机 $\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk}_{\mathrm{id}}}(\cdot)$ 获得私钥的泄露信息. $\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk}_{\mathrm{id}}}(\cdot)$ 的初始化参数包括安全参数 κ, 泄露参数 λ 和用户私钥 $\mathrm{sk}_{\mathrm{id}}$. $\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk}_{\mathrm{id}}}(\cdot)$ 收到来自敌手的高效可计算的泄露函数

$f_i : \{0,1\}^* \to \{0,1\}^{\lambda_i}$ 后, 返回 $\mathrm{sk_{id}}$ 的泄露 $f_i(\mathrm{sk_{id}})$ 给敌手, 但同一私钥 $\mathrm{sk_{id}}$ 的泄露总量不能超过 λ, 否则将输出无效符号 $\perp$. 特别地, 敌手能通过提交泄露函数 $f : \{0,1\}^* \to \{0,1\}^{\lambda}$ 询问一次 $\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk_{id}}}(\cdot)$, 并获得不超过 λ 的泄露 $f(\mathrm{sk_{id}})$ [17].

3.3.1 敌手 $\mathcal{A}^1$ 的抗泄露 CCA 安全性

若不存在敌手 $\mathcal{A}^1$ 在交互式实验 $\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda)$ 中以不可忽略的优势获胜, 则称 CL-KEM 在敌手 $\mathcal{A}^1$ 适应性选择密文攻击下具有不可区分性, 其中 $\mathcal{A}^1$ 的目标是判断挑战消息 k_β 是挑战身份 id^* 对应的封装密钥 k_1, 还是密钥空间 $\mathcal{K}$ 中选取的随机值 k_0.

在 $\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda)$ 中, 敌手 $\mathcal{A}^1$ 获胜的优势定义为

$$\mathsf{Adv}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda) = \left|\Pr\left[\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda)=1\right] - \Pr\left[\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda)=0\right]\right|,$$

其中概率来自随机数的使用.

$\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda)$:	$\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda)$:
$\mathrm{id}^* \leftarrow (\mathcal{A}^1)^{\mathcal{O}^{\mathsf{KeyGen}}(\cdot),\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk_{id}}}(\cdot),\mathcal{O}^{\mathsf{Decap}}(\cdot)}(\mathrm{Params})$;	$\mathrm{id}^* \leftarrow (\mathcal{A}^2)^{\mathcal{O}^{\mathsf{KeyGen}}(\cdot),\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk_{id}}}(\cdot),\mathcal{O}^{\mathsf{Decap}}(\cdot)}(\mathrm{Params},\mathrm{msk})$;
$(C^*,k_1)=\mathsf{Encap}(\mathrm{id}^*)$ and $k_0 \leftarrow_R \mathcal{K}$;	$(C^*,k_1)=\mathsf{Encap}(\mathrm{id}^*)$ and $k_0 \leftarrow_R \mathcal{K}$;
$\beta \leftarrow_R \{0,1\}$;	$\beta \leftarrow_R \{0,1\}$;
$\beta' \leftarrow (\mathcal{A}^1)^{\mathcal{O}^{\mathsf{KeyGen}}_{\neq \mathrm{id}^*}(\cdot),\mathcal{O}^{\mathsf{Decap}}_{\neq (C^*,\mathrm{id}^*)}(\cdot)}(\mathrm{Params},C^*,k_\beta)$;	$\beta' \leftarrow (\mathcal{A}^2)^{\mathcal{O}^{\mathsf{KeyGen}}_{\neq \mathrm{id}^*}(\cdot),\mathcal{O}^{\mathsf{Decap}}_{\neq (C^*,\mathrm{id}^*)}(\cdot)}(\mathrm{Params},\mathrm{msk},C^*,k_\beta)$;
如果 $\beta'=\beta$, 则输出 1; 否则输出 0.	如果 $\beta'=\beta$, 则输出 1; 否则输出 0.

其中 $\mathcal{K}$ 为封装密钥空间; $\mathcal{O}^{\lambda,\kappa}_{\mathrm{sk_{id}}}(\cdot)$ 是泄露谕言机, 敌手 ($\mathcal{A}^1$ 或 $\mathcal{A}^2$) 可获得任何身份对应私钥 $\mathrm{sk_{id}}$ 的泄露信息; $\mathcal{O}^{\mathsf{KeyGen}}(\cdot)$ 表示敌手向挑战者做任意身份的私钥和部分密钥生成询问; $\mathcal{O}^{\mathsf{KeyGen}}_{\neq \mathrm{id}^*}(\cdot)$ 表示敌手执行除挑战身份 id^* 之外其他身份的私钥和部分密钥生成询问; $\mathcal{O}^{\mathsf{Decap}}(\cdot)$ 表示敌手向提交关于身份密文对 (id,C) 的解封装询问; $\mathcal{O}^{\mathsf{Decap}}_{\neq (C^*,\mathrm{id}^*)}(\cdot)$ 表示敌手做除挑战身份和挑战密文对 (id^*,C^*) 以外的其他身份密文对 $(\mathrm{id},C)\neq(\mathrm{id}^*,C^*)$ 的解封装询问. 特别地, 敌手 $\mathcal{A}^2$ 由于掌握主私钥, 无需进行部分密钥生成询问.

3.3.2 敌手 $\mathcal{A}^2$ 的抗泄露 CCA 安全性

若不存在敌手 $\mathcal{A}^2$ 在交互式实验 $\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda)$ 中以不可忽略的优势获胜, 则称 CL-KEM 在敌手 $\mathcal{A}^2$ 适应性选择密文攻击下具有不可区分性.

在 $\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda)$ 中, 敌手 $\mathcal{A}^2$ 获胜的优势定义为

$$\mathsf{Adv}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda) = \left|\Pr\left[\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda)=1\right] - \Pr\left[\mathsf{Ext}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda)=0\right]\right|,$$

其中概率来自于随机数的使用.

定义3 (CL-KEM 泄露容忍的 CCA 安全性) 对于任意的敌手 $\mathcal{A}^1$ 和 $\mathcal{A}^2$, 若有 $\mathsf{Adv}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^1}(\kappa,\lambda)\leqslant \mathsf{negl}(\kappa)$ 和 $\mathsf{Adv}^{\text{LR-CCA}}_{\text{CL-KEM},\mathcal{A}^2}(\kappa,\lambda)\leqslant \mathsf{negl}(\kappa)$ 成立, 那么相应的 CL-KEM 具有泄露容忍的 CCA 安全性.

4 抗泄露的无证书密钥封装机制

本节给出抗泄露 CL-KEM 的具体实例 $\Pi=(\mathsf{Setup},\mathsf{KeyGen},\mathsf{Encap},\mathsf{Decap})$, 并在随机谕言机模型下基于 DDH 假设对安全性进行证明.

4.1 本文的构造

(1) $(\text{Params}, \text{msk}) \leftarrow \mathsf{Setup}(1^\kappa)$.

(i) 运行 $(q, G, P) \leftarrow \mathcal{G}(1^\kappa)$. 令 $H_1 : \{0,1\}^* \to Z_q^*$ 和 $H_2 : \{0,1\}^* \to Z_q^*$ 是两个密码学哈希函数; $\mathsf{KDF} : G \to Z_q^* \times Z_q^*$ 是安全的密钥衍射函数.

(ii) 随机选取主密钥 $\alpha \leftarrow_R Z_q^*$, 计算 $P_{\text{pub}} = \alpha P$, 并公开 $\text{Params} = \{q, G, P, P_{\text{pub}}, H_1, H_2, \mathsf{KDF}\}$.

(2) $(\text{pk}_{\text{id}}, \text{sk}_{\text{id}}) \leftarrow \mathsf{KeyGen}(\text{msk}, \text{id})$.

(i) 用户 U_{id} (身份标识为 id) 随机选取 $x_{\text{id}}^1, x_{\text{id}}^2 \in Z_q^*$, 计算 $X_{\text{id}} = (x_{\text{id}}^1 + x_{\text{id}}^2)P$, 发送 id 和 X_{id} 给 KGC;

(ii) KGC 随机选取 $r_{\text{id}} \in Z_q^*$, 计算 $Y_{\text{id}} = r_{\text{id}}P$ 和 $y_{\text{id}} = r_{\text{id}} + \alpha H_1(\text{id}, X_{\text{id}}, Y_{\text{id}})$, 然后, 将 y_{id} 和 Y_{id} 返回给用户 U_{id}, 其中 y_{id} 为部分私钥, Y_{id} 为部分公钥;

(iii) 通过验证 $y_{\text{id}}P = Y_{\text{id}} + P_{\text{pub}}H_1(\text{id}, X_{\text{id}}, Y_{\text{id}})$ 是否成立, 完成对 y_{id} 和 Y_{id} 的正确性验证. 则用户 U_{id} 的私钥是 $\text{sk}_{\text{id}} = (x_{\text{id}}^1, x_{\text{id}}^2, y_{\text{id}})$, 公钥是 $\text{pk}_{\text{id}} = (X_{\text{id}}, Y_{\text{id}})$.

(3) $(C, k) \leftarrow \mathsf{Encap}(\text{id}, \text{pk}_{\text{id}})$.

(i) 随机选取 $r, r_1, r_2 \in Z_q^*$, 并计算 $c_0 = rP$, $c_1 = r_1P$ 和 $c_2 = r_2P$;

(ii) 计算 $W = r_1X_{\text{id}} + r_2\mu(Y_{\text{id}} + P_{\text{pub}}H_1(\text{id}, X_{\text{id}}, Y_{\text{id}}))$, 其中 $\mu = H_2(c_0, c_1, c_2)$;

(iii) 计算 $c_3 = rt_1 + r_1t_2$, 其中 $(t_1, t_2) = \mathsf{KDF}(W)$;

(iv) 输出封装密文 $C = (c_0, c_1, c_2, c_3)$ 和封装密钥 $k = r_2X_{\text{id}} + r_1(Y_{\text{id}} + P_{\text{pub}}H_1(\text{id}, X_{\text{id}}, Y_{\text{id}}))$.

特别地, k 和 W 均由通用哈希函数生成, 由于该函数可视为特殊的随机性提取器, 确保封装密文中所有元素对任意敌手而言是随机的.

(4) $k \leftarrow \mathsf{Decap}(\text{sk}_{\text{id}}, C)$.

(i) 计算 $\mu = H_2(c_0, c_1, c_2)$ 和 $W' = (x_{\text{id}}^1 + x_{\text{id}}^2)c_1 + \mu y_{\text{id}}c_2$;

(ii) 计算 $(t_1', t_2') = \mathsf{KDF}(W')$, 若 $c_3P = t_1'c_0 + t_2'c_1$ 成立, 则输出 $k = (x_{\text{id}}^1 + x_{\text{id}}^2)c_2 + y_{\text{id}}c_1$; 否则输出 $\perp$.

由下述等式可获得上述基础 CL-KEM 实例的正确性.

$$\begin{aligned} r_1X_{\text{id}} + r_2\mu(Y_{\text{id}} + P_{\text{pub}}H_1(\text{id}, X_{\text{id}}, Y_{\text{id}})) &= r_1(x_{\text{id}}^1 + x_{\text{id}}^2)P + r_2\mu(r_{\text{id}} + \alpha H_1(\text{id}, X_{\text{id}}, Y_{\text{id}}))P \\ &= (x_{\text{id}}^1 + x_{\text{id}}^2)c_1 + \mu y_{\text{id}}c_2; \\ r_2X_{\text{id}} + r_1(Y_{\text{id}} + P_{\text{pub}}H_1(\text{id}, X_{\text{id}}, Y_{\text{id}})) &= r_2(x_{\text{id}}^1 + x_{\text{id}}^2)P + r_1(r_{\text{id}} + \alpha H_1(\text{id}, X_{\text{id}}, Y_{\text{id}}))P \\ &= (x_{\text{id}}^1 + x_{\text{id}}^2)c_2 + y_{\text{id}}c_1. \end{aligned}$$

4.2 安全性证明

对于任意的 $a, b, d \in Z_q^*$, 若 $abP = dP$, 则称 (P, aP, bP, dP) 是 DH 元组, 否则称其非 DH 元组.

定理2 对于任意的 $\lambda \leqslant 2\log q - \omega(\log\kappa)$, 若 DDH 假设成立, 那么上述 CL-KEM 实例具有泄露容忍的 CCA 安全性.

下面将通过两个引理完成对定理 1 的证明, 其中引理 1 表明在敌手 $\mathcal{A}^1$ 的攻击下本文实例具有泄露容忍的 CCA 安全性; 引理 2 表明在敌手 $\mathcal{A}^2$ 的攻击下本文实例具有泄露容忍的 CCA 安全性.

引理1 对于任意的 $\lambda \leqslant 2\log q - \omega(\log\kappa)$, 若存在一个敌手 $\mathcal{A}^1$ 在多项式时间内能以不可忽略的优势 $\mathsf{Adv}_{\text{CL-KEM},\mathcal{A}^1}^{\text{LR-CCA}}(\kappa, \lambda)$ 攻破本文实例泄露容忍的 CCA 安全性, 那么就能构造一个敌手 $\mathcal{B}$ 在多项式

时间内能以优势 $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{DDH}}(\kappa) \geqslant \frac{1}{e(Q_1+Q_2+1)}\mathsf{Adv}_{\text{CL-KEM},\mathcal{A}^1}^{\text{LR-CCA}}(\kappa,\lambda)$ 解决 DDH 假设的困难性, 其中 Q_1 是 $\mathcal{A}^1$ 提交的部分密钥生成询问的次数, Q_2 是 $\mathcal{A}^1$ 提交的私钥生成询问的次数, e 是自然对数底数.

证明 敌手 $\mathcal{B}$ 与敌手 $\mathcal{A}^1$ 进行泄露容忍的 CCA 安全性游戏之前, $\mathcal{B}$ 从 DDH 假设的挑战者处获得一个挑战元组 (P, aP, bP, dP) 及相应的公开参数 (q, G, P), 其中 $a, b, d \leftarrow_R Z_q^*$. $\mathcal{B}$ 的目标是当 $d = ab$ 时输出 1; 否则输出 0. $\mathcal{B}$ 维护 L_1, L_2, L_3 和 L_4 等列表用于记录相应的询问应答. $\mathcal{B}$ 与 $\mathcal{A}^1$ 间的交互过程如下所述:

(1) 初始化. $\mathcal{B}$ 运行 $(\text{Params}, \text{msk}) \leftarrow \mathsf{Setup}(1^\kappa)$, 在秘密保存 msk 的同时, 将 Params 发送给 $\mathcal{A}^1$.

(2) 阶段 1. 在挑战阶段之前, $\mathcal{B}$ 无法获知挑战身份, 因此 $\mathcal{B}$ 从 $\mathcal{A}^1$ 的询问中适应性的随机选取一个身份 id^* 作为挑战身份, 则 $\mathcal{B}$ 以概率 $\delta = \frac{1}{Q_1+Q_2+1}$ 猜中 $\mathcal{A}^1$ 提交的挑战身份, 其中 $Q_1 + Q_2 + 1$ 为游戏中 $\mathcal{A}^1$ 提交的身份总数. 该阶段 $\mathcal{A}^1$ 能适应性地进行多项式次下述询问.

(i) 部分密钥生成询问. 当收到 $\mathcal{A}^1$ 提出的部分密钥生成询问 $(\mathrm{id}, X_{\mathrm{id}})$ 时, $\mathcal{B}$ 进行下述操作:

若 $\mathrm{id} = \mathrm{id}^*$, 则返回终止符 $\perp$. 否则, 若 L_2 中存在以 $(\mathrm{id}, X_{\mathrm{id}})$ 为索引的记录 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}, y_{\mathrm{id}})$, 则返回 $(Y_{\mathrm{id}}, y_{\mathrm{id}})$ 给 $\mathcal{A}^1$; 若不存在, 则 $\mathcal{B}$ 随机选取 $y_{\mathrm{id}}, h_{\mathrm{id}} \leftarrow Z_q^*$ 且 $(*, *, *, h_{\mathrm{id}}) \notin L_1$, 并计算 $Y_{\mathrm{id}} = y_{\mathrm{id}}P - h_{\mathrm{id}}P_{\mathrm{pub}}$, 返回 $(Y_{\mathrm{id}}, y_{\mathrm{id}})$ 给 $\mathcal{A}^1$, 同时在 L_1 和 L_2 中分别添加 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}, h_{\mathrm{id}})$ 和 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}, y_{\mathrm{id}})$.

(ii) 随机谕言机 H_1 询问. 当 $\mathcal{B}$ 收到 $\mathcal{A}^1$ 提出的谕言机询问 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})$ 时, 返回 L_1 中以 id 为索引的相应记录 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}, h_{\mathrm{id}})$ 中的 h_{id} 给 $\mathcal{A}^1$. 特别地, $\mathcal{A}^1$ 进行随机谕言机询问之前需进行部分密钥生成询问, 确保 L_1 中肯定存在相应元组. 此外, H_2 是正常的密码学哈希函数.

(iii) 公钥生成询问. 当 $\mathcal{B}$ 收到 $\mathcal{A}^1$ 提出的公钥生成询问 (id) 时, 若 L_3 中存在以 id 为索引的记录, 则返回 $(X_{\mathrm{id}}, Y_{\mathrm{id}})$ 给 $\mathcal{A}^1$; 否则, $\mathcal{B}$ 执行下述操作:

若 $\mathrm{id} = \mathrm{id}^*$, 令 $X_{\mathrm{id}^*} = aP$ (隐含地设置 $x_{\mathrm{id}^*}^1 + x_{\mathrm{id}^*}^2 = a$); $\mathcal{B}$ 选取 $y_{\mathrm{id}^*}, h_{\mathrm{id}^*} \leftarrow Z_q^*$ 且满足 $(*, *, *, h_{\mathrm{id}^*}) \notin L_1$, 计算 $Y_{\mathrm{id}^*} = y_{\mathrm{id}^*}P - h_{\mathrm{id}^*}P_{\mathrm{pub}}$, 返回 $(X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$ 给 $\mathcal{A}^1$, 同时在 L_1, L_3 和 L_4 中分别添加相应的记录 $(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*}, h_{\mathrm{id}^*})$, $(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$ 和 $(\mathrm{id}^*, \perp, \perp, y_{\mathrm{id}^*})$. 对于 id^*, $\mathcal{B}$ 只掌握了 $\mathrm{sk}_{\mathrm{id}^*}$ 的部分内容 y_{id^*}.

若 $\mathrm{id} \neq \mathrm{id}^*$, 随机选取 $x_{\mathrm{id}^*}^1, x_{\mathrm{id}^*}^2 \leftarrow Z_q^*$, 并计算 $X_{\mathrm{id}^*} = (x_{\mathrm{id}^*}^1 + x_{\mathrm{id}^*}^2)P$; 然后以 $(\mathrm{id}^*, X_{\mathrm{id}^*})$ 作为输入运行部分密钥生成询问, 获得相应的应答 $(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*}, y_{\mathrm{id}^*})$, 返回 $(X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$ 给 $\mathcal{A}^1$, 同时在 L_3 和 L_4 中分别添加 $(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$ 和 $(\mathrm{id}^*, x_{\mathrm{id}^*}^1, x_{\mathrm{id}^*}^2, y_{\mathrm{id}^*})$.

(iv) 私钥生成询问. 当 $\mathcal{B}$ 收到 $\mathcal{A}^1$ 提出的私钥生成询问 (id) 时, 若 L_4 中存在以 id 为索引的记录, 则返回 $(x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$ 给 $\mathcal{A}^1$; 否则, $\mathcal{B}$ 执行下述操作:

若 $\mathrm{id} = \mathrm{id}^*$, 则返回终止符 $\perp$. 否则, 选取 $x_{\mathrm{id}}^1, x_{\mathrm{id}}^2 \leftarrow_R Z_q^*$, 并计算 $X_{\mathrm{id}} = (x_{\mathrm{id}}^1 + x_{\mathrm{id}}^2)P$; 然后以 $(\mathrm{id}, X_{\mathrm{id}})$ 作为输入运行部分密钥生成询问, 获得应答 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}, y_{\mathrm{id}})$, 返回 $(x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$ 给 $\mathcal{A}^1$ 的同时, 分别在列表 L_3 和 L_4 中添加记录 $(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})$ 和 $(\mathrm{id}, x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$.

(v) 公钥替换询问. $\mathcal{A}^1$ 将任意身份 id 的公钥 $\mathrm{pk}_{\mathrm{id}} = (X_{\mathrm{id}}, Y_{\mathrm{id}})$ 替换为自己掌握的 $pk'_{\mathrm{id}} = (X'_{\mathrm{id}}, Y'_{\mathrm{id}})$.

(vi) 解封装询问. 当收到 $\mathcal{A}^1$ 提出的解封装询问 (id, C) 时, $\mathcal{B}$ 分下述两类情况进行响应:

若 $\mathrm{id} = \mathrm{id}^*$, 以 id^* 为索引从 L_3 中查找记录 $(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$, 若存在 $r_1, r_2 \in Z_q^*$ 使得等式 $(t'_1, t'_2) = \mathsf{KDF}(r_1 X_{\mathrm{id}^*} + r_2\mu(Y_{\mathrm{id}^*} + P_{\mathrm{pub}}H_1(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})))$ 和 $c_3 P = t'_1 c_0 + t'_2 c_1$ 都成立, 其中 $\mu = H_2(c_0, c_1, c_2)$, 那么输出 $k = r_2 X_{\mathrm{id}^*} + r_1(Y_{\mathrm{id}^*} + P_{\mathrm{pub}}H_1(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*}))$.

若 $\mathrm{id} \neq \mathrm{id}^*$, 以 id 为索引从 L_4 中查找记录 $(\mathrm{id}, x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$, 然后以 $\mathrm{sk}_{\mathrm{id}} = (x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$ 和 C 作为输入运行解封装算法, 返回相应的结果 $k = \mathsf{Decap}(\mathrm{sk}_{\mathrm{id}}, C)$.

(vii) 泄露询问. 当收到 $\mathcal{A}^1$ 提出的泄露询问 $(\mathrm{id}, f_i : \{0,1\}^* \to \{0,1\}^{\lambda_i})$, 其中 $f_i : \{0,1\}^* \to \{0,1\}^{\lambda_i}$

是高效可计算的泄露函数, $\mathcal{B}$ 分下述两类情况进行响应:

若 $\mathrm{id} \neq \mathrm{id}^*$, 以 id 为索引从 L_4 中查找相应的记录 $(\mathrm{id}, x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$, 然后返回泄露 $f_i(\mathrm{sk}_{\mathrm{id}})$, 其中 $\mathrm{sk}_{\mathrm{id}} = (x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$, 但整个生命周期内敌手 $\mathcal{A}^1$ 获得的关于同一私钥的泄露总量不能超过 λ.

若 $\mathrm{id} = \mathrm{id}^*$, 以 id^* 为索引从 L_4 中查找相应的记录 $(\mathrm{id}^*, \perp, \perp, y_{\mathrm{id}^*})$, 然后返回泄露 $f_i(y_{\mathrm{id}^*})$. $\mathcal{A}^1$ 通过泄露询问仅获得用户私钥的部分信息, 因此返回 $f_i(y_{\mathrm{id}^*})$ 满足泄露函数的应答要求. 类似地, 关于挑战身份对应私钥的泄露总量同样不能超过 λ.

(3) 挑战. $\mathcal{A}^1$ 提交挑战身份 id_i 给 $\mathcal{B}$, 若 $\mathrm{id}_i \neq \mathrm{id}^*$, $\mathcal{B}$ 输出终止符 $\perp$; 否则, $\mathcal{B}$ 通过下述操作生成相应的挑战封装密文 $C^* = (c_0^*, c_1^*, c_2^*, c_3^*)$ 和封装秘钥 k_β:

(i) 以 id^* 为索引从 L_3 和 L_4 中分别查找记录 $(\mathrm{id}^*, X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$ 和 $(\mathrm{id}^*, \perp, \perp, y_{\mathrm{id}^*})$, 其中 $X_{\mathrm{id}^*} = aP$.

(ii) 令 $c_2^* = bP$ (隐含地设置 $r_2 = b$); 然后随机选取 $r, r_1 \leftarrow_R Z_q^*$, 并计算 $c_0^* = rP$ 和 $c_1^* = r_1 P$.

(iii) 计算 $W = r_1 X_{\mathrm{id}^*} + \mu y_{\mathrm{id}^*} c_2^*$, 其中 $\mu = H_2(c_0^*, c_1^*, c_2^*)$.

(iv) 计算 $c_3^* = rt_1 + r_1 t_2$ 和 $k_\beta = dP + y_{\mathrm{id}^*} c_1^*$, 其中 $(t_1, t_2) = \mathsf{KDF}(W)$.

最后输出挑战封装密文 $C^* = (c_0^*, c_1^*, c_2^*, c_3^*)$ 和封装密钥 k_β 给 $\mathcal{A}^1$. 当 $ab = d$ 时, $k_\beta = abP + y_{\mathrm{id}^*} c_1^* = (x_{\mathrm{id}^*}^1 + x_{\mathrm{id}^*}^2) c_2^* + y_{\mathrm{id}^*} c_1^*$, 则 k_β 是与 C^* 相对应的封装密钥; 当 $ab \neq d$ 时, 则 k_β 是群 G 上的随机元素.

(4) 阶段 2. 与阶段 1 相类似, $\mathcal{B}$ 响应 $\mathcal{A}^1$ 提出的相关询问. 但是, $\mathcal{A}^1$ 不能对挑战身份 id^* 进行私钥生成询问和部分密钥生成询问; 此外不能对挑战身份和挑战封装密文对 (id^*, C^*) 进行解封装询问. 特别地, 该阶段敌手不能提交任何的泄露询问.

(5) 输出. $\mathcal{A}^1$ 输出对 β 的判断. 若 $\beta = 1$, $\mathcal{B}$ 输出 1, 表示 (P, aP, bP, dP) 是一个 DH 元组, 否则, $\mathcal{B}$ 输出 0, 表示 (P, aP, bP, dP) 是一个非 DH 元组.

$\mathcal{A}^1$ 在询问阶段提交的部分密钥生成询问的次数是 Q_1, 提交的私钥生成询问的次数是 Q_2, 则整个游戏中共提交了 $Q_1 + Q_2 + 1$ 个身份. 令事件 $\mathcal{E}_1$ 表示 $\mathcal{B}$ 在询问阶段未终止; 事件 $\mathcal{E}_2$ 表示 $\mathcal{B}$ 在挑战阶段未终止, 因此 $\Pr[\mathcal{E}_1] = (1-\delta)^{Q_1+Q_2}$ 和 $\Pr[\mathcal{E}_1] = \delta$, 其中 $\delta = \frac{1}{Q_1+Q_2+1}$ 表示 $\mathcal{B}$ 猜中挑战身份 id^* 的概率. 则上述游戏中 $\mathcal{B}$ 未终止的概率是 $\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2] = (1-\delta)^{Q_1+Q_2}\delta \geqslant \frac{1}{e(Q_1+Q_2+1)}$, 其中 e 是自然对数底数.

$\mathcal{A}^1$ 的视图包括 C^*, 至多 λ 比特的泄露信息 Leak 和 Params, 其中 C^* 中的所有元素对于 $\mathcal{A}^1$ 而言都是随机的. 对于公钥 $\mathrm{pk}_{\mathrm{id}^*} = (X_{\mathrm{id}^*}, Y_{\mathrm{id}^*})$ 而言, $\mathcal{A}^1$ 无法从 X_{id^*} 获得相应 $x_{\mathrm{id}^*}^1$ 和 $x_{\mathrm{id}^*}^2$ 的信息, 因为 X_{id^*} 对应多个 $x_{\mathrm{id}^*}^1$ 和 $x_{\mathrm{id}^*}^2$ 的组合, 主私钥的保密性确保 $\mathcal{A}^1$ 无法从 Y_{id^*} 获得 y_{id^*} 的信息. 由剩余哈希引理 [15] 可知 $\widetilde{H}_\infty(\mathrm{sk}_{\mathrm{id}^*} | \mathrm{pk}_{\mathrm{id}^*}, C^*, \mathrm{Params}, \mathsf{Leak}) = \widetilde{H}_\infty((x_{\mathrm{id}^*}^1, x_{\mathrm{id}^*}^2, y_{\mathrm{id}^*}) | \mathsf{Leak}) \geqslant 3\log q - \lambda$. 由于封装密钥是由通用哈希函数生成, 并且 $\mathrm{sk}_{\mathrm{id}^*}$ 经泄露和运算损耗后剩余的平均最小熵不能小于通用哈希函数输出值应具有的最小熵, 那么有 $\log q \leqslant 3\log q - \lambda - \omega(\log\kappa)$, 因此 $\lambda \leqslant 2\log q - \omega(\log\kappa)$, 其中 $\omega(\log\kappa)$ 表示运算过程 $\mathrm{sk}_{\mathrm{id}^*}$ 的额外运算损耗信息.

综上所述, 对于 $\lambda \leqslant 2\log q - \omega(\log\kappa)$, 若 $\mathcal{A}^1$ 能以不可忽略的优势 $\mathsf{Adv}_{\text{CL-KEM},\mathcal{A}^1}^{\text{LR-CCA}}(\kappa, \lambda)$ 攻破本文实例泄露容忍的 CCA 安全性, 则 $\mathcal{B}$ 能以显而易见的优势 $\mathsf{Adv}_{\mathcal{B}}^{\text{DDH}}(\kappa) \geqslant \frac{1}{e(Q_1+Q_2+1)}\mathsf{Adv}_{\text{CL-KEM},\mathcal{A}^1}^{\text{LR-CCA}}(\kappa, \lambda)$ 攻破经典的 DDH 困难性假设.

引理2 对于任意的 $\lambda \leqslant 2\log q - \omega(\log\kappa)$, 若存在一个敌手 $\mathcal{A}^2$ 在多项式时间内能以不可忽略的优势 $\mathsf{Adv}_{\text{CL-KEM},\mathcal{A}^2}^{\text{LR-CCA}}(\kappa, \lambda)$ 攻破本文实例泄露容忍的 CCA 安全性, 那么就能构造一个敌手 $\mathcal{B}$ 在多项式时间内能以优势 $\mathsf{Adv}_{\mathcal{B}}^{\text{DDH}}(\kappa) \geqslant \frac{1}{e(Q_2+1)}\mathsf{Adv}_{\text{CL-KEM},\mathcal{A}^1}^{\text{LR-CCA}}(\kappa, \lambda)$ 解决 DDH 假设的困难性.

引理 1 的证明中, 挑战元组 (P, aP, bP, dP) 的元素并未嵌入主私钥, 确保引理 1 的证明中, $\mathcal{B}$ 持有完整的主私钥, 因此可以使用引理 1 的证明思路对引理 2 进行证明, 其中 $\mathcal{B}$ 具备将主私钥发送 $\mathcal{A}^2$ 的

能力, 此处不再赘述详细的证明过程. 特别地, 在引理 2 的证明中, $\mathcal{A}^2$ 无需进行部分密钥生成询问.

5 抵抗连续泄露攻击的无证书密钥封装机制

在实际环境中, 敌手能通过边信道、冷启动等各种各样的泄露攻击对密码机制进行持续攻击, 因此为增强密码机制的实用性, 需进一步研究密码机制抗连续泄露攻击的能力. 当下述两个条件都成立时, 可通过定期更新用户私钥的方法在有界泄露密码机制的基础上获得连续泄露容忍性 [16]: (1) 私钥更新过程中, 密码机制的公开参数保持不变, 且功能不会发生改变; (2) 对于敌手而言, 密钥更新算法更新后的用户私钥与原始私钥是不可区分的. 基于上述结论本节将在上文实例 II 的基础上设计抵抗连续泄露攻击的 CL-KEM.

5.1 具体构造

抗连续泄露 CL-KEM 实例 $\Pi' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Encap}', \mathsf{Decap}', \mathsf{Update}')$ 主要包含下述 5 个算法, 其中 Update' 是附加的密钥更新算法, 定期对用户私钥进行更新.

(1) 算法 $(\mathrm{Params}, \mathrm{msk}) \leftarrow \mathsf{Setup}'(1^\kappa)$ 与基础 CL-KEM 实例 II 的初始化算法 Setup 一样.

(2) $(\mathrm{pk_{id}}, \mathrm{sk_{id}}) \leftarrow \mathsf{KeyGen}(\mathrm{msk}, \mathrm{id})$.

(i) 用户 U_{id} (身份标识为 id) 随机选取 $x_{\mathrm{id}}^1, x_{\mathrm{id}}^2 \in Z_q^*$, 计算 $X_{\mathrm{id}} = (x_{\mathrm{id}}^1 + x_{\mathrm{id}}^2)P$, 发送 id 和 X_{id} 给 KGC;

(ii) KGC 随机选取 $r_{\mathrm{id}} \in Z_q^*$, 计算 $Y_{\mathrm{id}} = r_{\mathrm{id}}P$ 和 $y_{\mathrm{id}} = r_{\mathrm{id}} + \alpha H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})$, 然后, 将 y_{id} 和 Y_{id} 返回给用户, 其中 y_{id} 为部分私钥, Y_{id} 为部分公钥; 通过验证 $y_{\mathrm{id}}P = Y_{\mathrm{id}} + P_{\mathrm{pub}}H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})$ 是否成立, 完成对 y_{id} 和 Y_{id} 的正确性验证;

(iii) 用户 U_{id} 随机选取 $\boldsymbol{w} = (w_1, w_2, w_3) \in (Z_q^*)^3$, 并计算 $\boldsymbol{d} = (d_1, d_2, d_3) = (x_{\mathrm{id}}^1 - w_1, x_{\mathrm{id}}^2 - w_2, y_{\mathrm{id}} - w_3)$. 那么用户 U_{id} 的私钥是 $\mathrm{sk_{id}} = (\boldsymbol{d}, \boldsymbol{w})$, 公钥是 $\mathrm{pk_{id}} = (X_{\mathrm{id}}, Y_{\mathrm{id}})$.

(3) $\mathrm{sk}'_{\mathrm{id}} \leftarrow \mathsf{Update}'(\mathrm{sk_{id}})$.

(i) 随机选取 $\boldsymbol{n} = (n_1, n_2, n_3) \in (Z_q^*)^3$, 并计算 $\boldsymbol{d}' = \boldsymbol{d} + \boldsymbol{n}$ 和 $\boldsymbol{w}' = \boldsymbol{w} - \boldsymbol{n}$;

(ii) 输出更新后的私钥 $\mathrm{sk}'_{\mathrm{id}} = (\boldsymbol{d}', \boldsymbol{w}')$. 对于任意时刻执行的密钥更新算法, 更新后的私钥 $\mathrm{sk}'_{\mathrm{id}}$ 与原始私钥 $\mathrm{sk_{id}}$ 满足 $\mathrm{sk}'_{\mathrm{id}} \neq \mathrm{sk_{id}}$ 和 $|\mathrm{sk}'_{\mathrm{id}}| = |\mathrm{sk_{id}}|$. 此外, 随机向量 $\boldsymbol{n}$ 的使用使得 $\mathrm{sk}'_{\mathrm{id}}$ 和 $\mathrm{sk_{id}}$ 对任意敌手都是不可区分的.

(4) 算法 $(C, k) \leftarrow \mathsf{Encap}'(\mathrm{id})$ 与基础 CL-KEM 实例 II 的封装算法 Encap 一样.

(5) $k \leftarrow \mathsf{Decap}(\mathrm{sk_{id}}, C)$.

(i) 计算 $(x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}}) = \boldsymbol{d} + \boldsymbol{w}$;

(ii) 计算 $W' = (x_{\mathrm{id}}^1 + x_{\mathrm{id}}^2)c_1 + \mu y_{\mathrm{id}}c_2$ 和 $(t_1', t_2') = \mathsf{KDF}(W')$, 其中 $\mu = H_2(c_0, c_1, c_2)$;

(iii) 若 $c_3P = t_1'c_0 + t_2'c_1$ 成立, 则输出 $k = (x_{\mathrm{id}}^1 + x_{\mathrm{id}}^2)c_2 + y_{\mathrm{id}}c_1$; 否则输出 $\perp$.

由于密钥更新算法并未改变底层的核心秘密 $(x_{\mathrm{id}}^1, x_{\mathrm{id}}^2, y_{\mathrm{id}})$, 因此由第 4 节基础 CL-KEM 实例 II 的正确性分析可知, 本节抗连续泄露 CL-KEM 实例是正确的. 实质上, 上述机制通过多对一的映射关系完成对核心秘密的替代, 进而实现抵抗连续泄露攻击的目的, 并且方案的安全性是由底层核心秘密所决定, 用户私钥是底层核心秘密的替代形式, 也就是说用户私钥空间中的每个值通过映射关系能够恢复出方案的核心秘密值. 密钥更新操作完成用户私钥空间中各元组间的转变, 并且多对一保证了用户更新私钥的正确性.

5.2 安全性分析

由于连续泄露的最大优势是可通过定期执行密钥更新算法将连续泄露的问题转化为有界泄露容忍性进行研究, 本文抵抗连续泄露攻击的 CL-KEM 实例是在第 4 节机制的基础之上设计的, 因此该机制的正确性和安全性可由底层 CL-KEM 的相应性质获得; 而该机制的连续泄露容忍性可由密钥更新算法和底层 CL-KEM 的有界泄露容忍性获得.

下面我们主要对泄露参数进行详细的分析: 敌手的视图包括封装密文, 至多 λ 比特的私钥泄露信息 Leak 和系统公开参数 Params, 其中封装密文中的所有元素对于敌手而言都是均匀随机的, 则敌手无法从封装密文和公开参数中获知用户私钥的泄露信息. 对于公钥 $\mathrm{pk}_{\mathrm{id}^*}=(X_{\mathrm{id}^*},Y_{\mathrm{id}^*})$ 而言, 敌手无法从 X_{id^*} 获得相应 $\boldsymbol{d}$ 和 $\boldsymbol{w}$ 的信息, 因为一个公钥 $\mathrm{pk}_{\mathrm{id}^*}$ 对应多个 $\boldsymbol{d}$ 和 $\boldsymbol{w}$ 的组合, 增强了私钥的随机性. 由定理 2 的证明可知 $\widetilde{H}_\infty(\mathrm{sk}_{\mathrm{id}^*}|\mathrm{pk}_{\mathrm{id}^*},C^*,\mathrm{Params},\mathsf{Leak})=\widetilde{H}_\infty((\boldsymbol{d},\boldsymbol{w})|\mathsf{Leak})\geqslant 6\log q-\lambda$. 由于封装密钥是由通用哈希函数生成的, 则有 $\log q\leqslant 6\log q-\lambda-\omega(\log\kappa)$, 因此 $\lambda\leqslant 5\log q-\omega(\log\kappa)$.

综上所述, 对于任意的 $\lambda\leqslant 5\log q-\omega(\log\kappa)$, 若敌手能以不可忽略的优势攻破实例 Π' 泄露容忍的 CCA 安全性, 那么能构造一个敌手以显而易见的优势解决 DDH 假设的困难性.

6 抗泄露性能最佳的无证书密钥封装机制

在抗泄露密码学中, 泄漏率 $l_{\mathrm{leak}}=\frac{\lambda}{l_{\mathrm{sk}}}$ 是指允许的最大泄露长度 λ 与私钥长度 l_{sk} 的比值, 其中 $\lambda\leqslant l_{\mathrm{sk}}-\mathsf{Fun}(\kappa)$, $\mathsf{Fun}(\kappa)$ 表示计算过程中的泄露损耗. 为了达到最佳的泄露率, 在保持泄露损耗 $\mathsf{Fun}(\kappa)$ 和公开参数形式不变的情况下, 尽最大可能去增加私钥的长度 l_{sk}, 当 l_{sk} 足够大时, 泄露率即可达到 $1-\mathcal{O}(1)$. 针对上述需求, 本节设计抗泄露性能最佳的 CL-KEM.

6.1 具体构造

CL-KEM 的新型实例 $\Pi''=(\mathsf{Setup}'',\mathsf{KeyGen}'',\mathsf{Encap}'',\mathsf{Decap}'')$ 具体包含下述 4 个算法:

(1) 算法 $(\mathrm{Params},\mathrm{msk})\leftarrow\mathsf{Setup}''(1^\kappa)$ 与基础 CL-KEM 实例 Π 的初始化算法 Setup 一样.

(2) $(\mathrm{pk}_{\mathrm{id}},\mathrm{sk}_{\mathrm{id}})\leftarrow\mathsf{KeyGen}''(\mathrm{msk},\mathrm{id})$.

(i) 对于私钥元素个数 $n\in\mathbb{N}$, 用户 U_{id} 随机选取 $\boldsymbol{x}_{\mathrm{id}}=(x_{\mathrm{id}}^1,x_{\mathrm{id}}^2,\ldots,x_{\mathrm{id}}^n)\in(Z_q^*)^n$, 计算 $X_{\mathrm{id}}=(\sum_{i=1}^n x_{\mathrm{id}}^i)P$, 发送 id 和 X_{id} 给 KGC;

(ii) KGC 随机选取 $r_{\mathrm{id}}^1,r_{\mathrm{id}}^2,\ldots,r_{\mathrm{id}}^n\leftarrow_R Z_q^*$, 计算

$$Y_{\mathrm{id}}=\left(\sum_{i=1}^n r_{\mathrm{id}}^i\right)P \text{ 和 } y_{\mathrm{id}}^i=r_{\mathrm{id}}^i+\alpha H_1(\mathrm{id},X_{\mathrm{id}},Y_{\mathrm{id}})_{i=1,2,\ldots,n},$$

然后, 将 $\boldsymbol{y}_{\mathrm{id}}=(y_{\mathrm{id}}^1,y_{\mathrm{id}}^2,\ldots,y_{\mathrm{id}}^n)$ 和 Y_{id} 返回给用户, 其中 $\boldsymbol{y}_{\mathrm{id}}$ 为部分私钥, Y_{id} 为部分公钥;

(iii) 通过验证 $(\sum_{i=1}^n y_{\mathrm{id}}^i)P=Y_{\mathrm{id}}+nP_{\mathrm{pub}}H_1(\mathrm{id},X_{\mathrm{id}},Y_{\mathrm{id}})$ 是否成立, 完成对 $\boldsymbol{y}_{\mathrm{id}}$ 和 Y_{id} 的正确性验证. 则用户 U_{id} 的公钥是 $\mathrm{pk}_{\mathrm{id}}=(X_{\mathrm{id}},Y_{\mathrm{id}})$, 私钥是 $\mathrm{sk}_{\mathrm{id}}=(\boldsymbol{x}_{\mathrm{id}},\boldsymbol{y}_{\mathrm{id}})$.

(3) 算法 $(C,k)\leftarrow\mathsf{Encap}''(\mathrm{id})$ 的主要操作有:

(i) 随机选取 $r,r_1,r_2\in Z_q^*$, 并计算 $c_0=rP$, $c_1=r_1P$ 和 $c_2=r_2P$;

(ii) 计算 $W=r_1X_{\mathrm{id}}+r_2\mu(Y_{\mathrm{id}}+nP_{\mathrm{pub}}H_1(\mathrm{id},X_{\mathrm{id}},Y_{\mathrm{id}}))$, 其中 $\mu=H_2(c_0,c_1,c_2)$;

(iii) 计算 $c_3=rt_1+r_1t_2$, 其中 $(t_1,t_2)=\mathsf{KDF}(W)$.

最后输出封装密文 $C=(c_0,c_1,c_2,c_3)$ 和封装密钥 $k=r_2X_{\mathrm{id}}+r_1(Y_{\mathrm{id}}+nP_{\mathrm{pub}}H_1(\mathrm{id},X_{\mathrm{id}},Y_{\mathrm{id}}))$.

(4) $k\leftarrow\mathsf{Decap}''(\mathrm{sk}_{\mathrm{id}},C)$.

(i) 计算 $\mu = H_2(c_0, c_1, c_2)$ 和 $W' = \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_1 + \mu\left(\sum_{i=1}^{n} y_{\mathrm{id}}^i\right)c_2$;

(ii) 计算 $(t_1', t_2') = \mathsf{KDF}(W')$, 若 $c_3 P = t_1' c_0 + t_2' c_1$ 成立, 则输出 $k = \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_2 + \left(\sum_{i=1}^{n} y_{\mathrm{id}}^i\right)c_1$; 否则输出 $\perp$.

由下述等式可获得本文 CL-KEM 新型实例 Π'' 的正确性.

$$
\begin{aligned}
r_1 X_{\mathrm{id}} + r_2\mu(Y_{\mathrm{id}} + nP_{\mathrm{pub}}H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})) &= r_1\left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)P + r_2\mu\left(\left(\sum_{i=1}^{n} r_{\mathrm{id}}^i\right)P + n\alpha H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})P\right)\\
&= \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_1 + \mu\left(\left(\sum_{i=1}^{n} r_{\mathrm{id}}^i\right) + n\alpha H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})\right)c_2\\
&= \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_1 + \mu\left(\sum_{i=1}^{n}(r_{\mathrm{id}}^i + \alpha H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}))\right)c_2\\
&= \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_1 + \mu\left(\sum_{i=1}^{n} y_{\mathrm{id}}^i\right)c_2,\\
r_2 X_{\mathrm{id}} + r_1(Y_{\mathrm{id}} + nP_{\mathrm{pub}}H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})) &= r_2\left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)P + r_1\left(\left(\sum_{i=1}^{n} r_{\mathrm{id}}^i\right) + n\alpha H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}})\right)P\\
&= \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_2 + \left(\sum_{i=1}^{n}(r_{\mathrm{id}}^i + \alpha H_1(\mathrm{id}, X_{\mathrm{id}}, Y_{\mathrm{id}}))\right)c_1\\
&= \left(\sum_{i=1}^{n} x_{\mathrm{id}}^i\right)c_2 + \left(\sum_{i=1}^{n} y_{\mathrm{id}}^i\right)c_1.
\end{aligned}
$$

6.2 安全性分析

定理3 对于任意的 $\lambda \leqslant (2n-1)\log q - \omega(\log\kappa)$, 若 DDH 困难性假设成立, 那么本文 CL-KEM 实例 Π'' 具有泄露容忍的 CCA 安全性; 且当参数 n 足够大时, 该实例的泄露率将达到 $1 - \mathcal{O}(1)$.

定理 3 的证明过程与定理 2 相类似, 此处不再赘述. 下面详细分析泄露参数: 敌手的视图包括封装密文, 至多 λ 比特的私钥泄露信息和系统公开参数 Leak, 其中封装密文中的所有元素对于敌手而言都是均匀随机的, 则敌手无法从封装密文和公开参数中获知用户私钥的泄露信息. 对于公钥 $\mathrm{pk}_{\mathrm{id}} = (X_{\mathrm{id}}, Y_{\mathrm{id}})$ 而言, 敌手无法从 X_{id} 和 Y_{id} 获得相应 $\boldsymbol{x}_{\mathrm{id}}$ 和 $\boldsymbol{y}_{\mathrm{id}}$ 的信息, 因为一个公钥 $\mathrm{pk}_{\mathrm{id}}$ 将对应多个 $\boldsymbol{x}_{\mathrm{id}}$ 和 $\boldsymbol{y}_{\mathrm{id}}$ 的组合.

由定理 2 的证明可知 $\widetilde{H}_\infty(\mathrm{sk}_{\mathrm{id}^*}|\mathrm{pk}_{\mathrm{id}^*}, C^*, \mathrm{Params}, \mathsf{Leak}) = \widetilde{H}_\infty((\boldsymbol{x}_{\mathrm{id}}, \boldsymbol{y}_{\mathrm{id}})|\mathsf{Leak}) \geqslant 2n\log q - \lambda$, 那么有 $\log q \leqslant 2n\log q - \lambda - \omega(\log\kappa)$, 可知 $\lambda \leqslant (2n-1)\log q - \omega(\log\kappa)$. 那么当 n 足够大时, 有 $\frac{\lambda}{l_{\mathrm{sk}}} \leqslant \frac{(2n-1)\log q - \omega(\log\kappa)}{2n\log q} \approx 1 - \mathcal{O}(1)$.

综上所述, 在本节 CL-KEM 的新型构造中, 当参数 n 足够大时, 其泄露率达到 $1 - \mathcal{O}(1)$.

7 抗泄露无证书密钥封装机制的应用

本节基于抗泄露 CL-KEM 分别设计抗泄露无证书混合加密机制和抗泄露无证书密钥协商协议的通用构造, 并且下述通用构造的抗泄露性均来自底层 CL-KEM 的泄露容忍性. 除此之外, 抗泄露 CL-KEM 还有其他一些应用, 例如以该原语为底层工具构造云计算的数据授权机制, 达到抗泄露攻击的数据授权访问目的.

7.1 抗泄露的无证书混合加密机制

混合加密通过对称加密和公钥加密的优势互补实现了快速而安全的加密运算. 因此混合加密机制由密钥封装机制 (key encapsulation mechanism, KEM) 和数据封装机制 (data encapsulation mechanism, DEM) 两部分组成, 其中 KEM 是公钥密码机制, DEM 是对称密码机制.

令基础机制 $\Pi_1 = (\mathsf{KEM.Setup}, \mathsf{KEM.KeyGen}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap})$ 是泄露容忍 CCA 安全的 CL-KEM, $\Pi_2 = (\mathsf{DEM.Enc}, \mathsf{DEM.Dec})$ 是 CCA 安全的数据封装机制, 那么抵抗泄露攻击的无证书混合加密机制 $\Pi_3 = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ 的通用构造由下述算法组成:

(1) $(\text{Params}, \text{msk}) \leftarrow \mathsf{Setup}(1^\kappa)$.

输出 Params 和 msk, 其中 $(\text{Params}, \text{msk}) \leftarrow \mathsf{KEM.Setup}(1^\kappa)$.

(2) $(\text{pk}_{\text{id}}, \text{sk}_{\text{id}}) \leftarrow \mathsf{KeyGen}(\text{id}, \text{msk})$.

输出身份 $\text{id} \in \mathcal{ID}$ 的公钥 pk_{id} 和私钥 sk_{id}, 其中 $(\text{pk}_{\text{id}}, \text{sk}_{\text{id}}) \leftarrow \mathsf{KEM.KeyGen}(\text{id}, \text{msk})$.

(3) $C \leftarrow \mathsf{Enc}(\text{id}, \text{pk}_{\text{id}}, M)$.

计算 $(c_1, k) = \mathsf{KEM.Encap}(\text{pk}_{\text{id}}, \text{id})$ 和 $c_2 = \mathsf{DEM.Enc}(k, M)$. 输出 $C = (c_1, c_2)$.

(4) $M \leftarrow \mathsf{Dec}(\text{sk}_{\text{id}}, C)$.

若 $\mathsf{KEM.Decap}(\text{sk}_{\text{id}}, c_1) = \perp$, 则终止; 否则计算 $k = \mathsf{KEM.Decap}(\text{sk}_{\text{id}}, c_1)$ 和 $M = \mathsf{DEM.Dec}(k, c_2)$, 最后输出 M.

定理4 若底层的 Π_1 是泄露容忍 CCA 安全的 CL-KEM, Π_2 是 CCA 安全的数据封装机制, 那么上述通用构造是泄露容忍 CCA 安全的无证书混合加密机制.

文献 [18] 指出 CCA 安全的密钥封装机制和 CCA 安全的数据封装机制能够构造 CCA 安全的混合加密机制. 由上述结论可知由 CCA 安全的 CL-KEM 和 CCA 安全的 DEM 即可得到 CCA 安全的无证书混合加密机制, 唯一的区别是本文构造增加了抵抗泄露攻击的能力.

7.2 抗泄露的无证书密钥协商协议

密钥协商是指两个实体共同建立会话密钥的过程, 并且任何一个参与者均对结果产生影响, 同时密钥协商过程不需要任何可信的第三方.

令 $\Pi_1 = (\mathsf{KEM.Setup}, \mathsf{KEM.KeyGen}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap})$ 是泄露容忍 CCA 安全的 CL-KEM, 其对应的封装密钥空间为 $\mathcal{K} = Z_q^*$; $\mathsf{KDF}: \{0,1\}^* \rightarrow \{0,1\}^{l_k}$ 是安全的密钥衍射函数, 那么抗泄露无证书密钥协商协议具体包含下述两个阶段:

(1) 初始化阶段.

参与者 Alice (身份标识为 id_A) 和 Bob (身份标识为 id_B) 分别运行

$$(\text{pk}_A, \text{sk}_A) \leftarrow \mathsf{KEM.KeyGen}(\text{id}_A, \text{msk}) \text{ 和 } (\text{pk}_B, \text{sk}_B) \leftarrow \mathsf{KEM.KeyGen}(\text{id}_B, \text{msk}).$$

特别地, 该协议需要第三方 KGC 进行系统初始化, 生成相应的系统公开参数 Params 和主私钥 msk.

(2) 密钥协商阶段.

(i) Alice 随机选取 $x_A \leftarrow_R Z_q^*$, 并计算 $X_A = g^{x_A}$ 和 $(c_A, k_A) = \mathsf{KEM.Encap}(\text{pk}_{\text{id}_B}, \text{id}_B)$, 并发送消息 (X_A, c_A) 给 Bob;

(ii) Bob 随机选取 $y_B \leftarrow_R Z_q^*$, 并计算 $Y_B = g^{y_B}$ 和 $(c_B, k_B) = \mathsf{KEM.Encap}(\text{pk}_{\text{id}_A}, \text{id}_A)$, 并发送消息 (X_B, c_B) 给 Alice;

(iii) Alice 计算 $k'_B = \mathsf{KEM.Decap}(\text{sk}_{\text{id}_A}, c_B)$, 并输出协商密钥 $k = \mathsf{KDF}(\text{id}_A, \text{id}_B, g^{k_A k'_B}, Y_B^{x_A})$; Bob 计算 $k'_A = \mathsf{KEM.Decap}(\text{sk}_{\text{id}_B}, c_A)$, 并输出协商密钥 $k = \mathsf{KDF}(\text{id}_A, \text{id}_B, g^{k'_A k_B}, X_A^{y_B})$.

定理5 若底层的 Π_1 是泄露容忍的 CCA 安全的 CL-KEM, KDF 是安全的密钥衍射函数, 那么本文上述通用构造是抗泄露的无证书密钥协商协议.

文献 [15] 在身份基哈希证明系统的基础上设计了抗泄露的身份基密钥协商协议, 由于该机制的底层工具是基于身份的密钥封装机制, 因此可借鉴该方法对本文抗泄露无证书密钥协商协议的安全性进行证明, 区别是本文密钥协商协议中使用的是无证书的密钥封装机制.

8 结论

为提升抗泄露 CL-KEM 的计算效率和泄露容忍性, 本文在不使用双线性映射的前提下, 分别设计了抗泄露攻击的 CL-KEM 和抗连续泄露攻击的 CL-KEM, 同时基于 DDH 困难性假设对相应构造的安全性进行了形式化证明; 此外, 封装密文的所有元素对于任意的敌手而言是随机的, 确保敌手无法从封装密文中获得泄露信息; 同时保持泄露参数是固定常数, 与封装密钥空间的大小无关. 为进一步获得更优的泄露容忍性, 在保持公开参数形式不变的前提下, 通过增加用户私钥的长度, 设计了一个新型的抗泄露 CL-KEM, 该机制所容忍的泄露长度将达到 $l_{\mathrm{sk}}(1-\mathcal{O}(1))$, 其中 l_{sk} 表示私钥长度. 最后, 本文基于抗泄露 CL-KEM 设计了抗泄露无证书混合加密机制和抗泄露无证书密钥协商协议的通用构造, 扩展了抗泄露 CL-KEM 的应用范围.

由于标准模型下密码机制的安全性更佳, 下一阶段我们将研究标准模型下抗泄露 CL-KEM 的具体构造. 此外, 为解决抗泄露能力的固定不变性, 我们还将研究能够动态调整抗泄露能力的 CL-KEM, 根据实际应用环境的需求, 密钥封装算法自适应的调节 CL-KEM 的抗泄露能力.

参考文献

1 Al-Riyami S S, Paterson K G. Certificateless public key cryptography. In: Proceedings of the 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, 2003. 452–473

2 Tedeschi P, Sciancalepore S, Eliyan A, et al. LiKe: lightweight certificateless key agreement for secure IoT communications. IEEE Int Things J, 2020, 7: 621–638

3 Wu G, Zhang F T, Shen L M, et al. Certificateless aggregate signature scheme secure against fully chosen-key attacks. Inf Sci, 2020, 514: 288–301

4 Cheng Z H, Chen L Q, Ling L, et al. General and efficient certificateless public key encryption constructions. In: Proceedings of International Conference on Pairing-Based Cryptography, Tokyo, 2007. 83–107

5 Huang Q, Wong D S. Generic certificateless key encapsulation mechanism. In: Proceedings of the 12th Australasian Conference on Information Security and Privacy, Townsville, 2007. 215–229

6 Zhou Y W, Yang B, Cheng H, et al. Aleakage-resilient certificateless public key encryption scheme with CCA2 security. Front Inf Technol Electron Eng, 2018, 19: 481–493

7 Zhou Y W, Yang B. Leakage-resilient CCA2-secure certificateless public-key encryption scheme without bilinear pairing. Inf Process Lett, 2018, 130: 16–24

8 Zhou Y W, Yang B. Continuous leakage-resilient certificateless public key encryption with CCA security. Knowl-Based Syst, 2017, 136: 27–36

9 Wu J D, Tseng Y M, Huang S S, et al. Leakage-resilient certificateless key encapsulation scheme. Informatica, 2018, 29: 125–155

10 Zhou Y W, Yang B, Xia Z, et al. Revocable identity-based encryption scheme with leakage-resilience. Chinese J Comput, 2020, 43: 1534–1554 [周彦伟, 杨波, 夏喆, 等. 抵抗泄露攻击的可撤销 IBE 机制. 计算机学报, 2020, 43: 1534–1554]

11 Chow S S M, Dodis Y, Rouselakis Y, et al. Practical leakage-resilient identity-based encryption from simple assumptions. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, Chicago, 2010. 152–161

12 Zhou Y W, Yang B, Zhang W Z. Provably secure and efficient leakage-resilient certificateless signcryption scheme without bilinear pairing. Discrete Appl Math, 2016, 204: 185–202
13 Liu S L, Weng J, Zhao Y L. Efficient public key cryptosystem resilient to key leakage chosen ciphertext attacks. In: Proceedings of Cryptographers' Track at the RSA Conference, San Francisco, 2013. 84–100
14 Dodis Y, Haralambiev K, López-Alt A, et al. Efficient public-key cryptography in the presence of key leakage. In: Proceedings of the 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, 2010. 613–631
15 Ruan O, Zhang Y Y, Zhang M, et al. After-the-fact leakage-resilient identity-based authenticated key exchange. IEEE Syst J, 2018, 12: 2017–2026
16 Dodis Y, Haralambiev K, López-Alt A, et al. Cryptography against continuous memory attacks. In: Proceedings of the 51th Annual IEEE Symposium on Foundations of Computer Science, Las Vegas, 2010. 511–520
17 Joël A, Dodis Y, Naor M, et al. Public-key encryption in the bounded-retrieval model. In: Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco, 2010. 113–134
18 Qin B D, Liu S L, Chen K F. Efficient chosen-ciphertext secure public-key encryption scheme with high leakage-resilience. IET Inf Secur, 2015, 9: 32–42

Leakage-resilient certificateless key-encapsulation mechanism and application

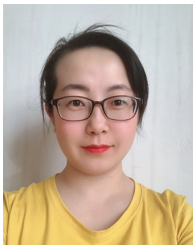
Yanwei ZHOU[1,2,4], Bo YANG[1*], Zirui QIAO[1*], Zhe XIA[3] & Mingwu ZHANG[2,4]

1. *School of Computer Science, Shaanxi Normal University, Xi'an* 710062*, China*;
2. *Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, Guilin* 541004*, China*;
3. *School of Computer Science and Technology, Wuhan University of Technology, Wuhan* 430070*, China*;
4. *State Key Laboratory of Cryptology, Beijing* 100878*, China*
* Corresponding author. E-mail: byang@snnu.edu.cn, qzr_snnu@163.com

Abstract Certificateless public key cryptography which has attracted great interest can solve the certificate management issue of the traditional public-key cryptography system, at the same time, can also avoid the key escrow in identity-based cryptography. We assume that the adversary cannot obtain the leakage information of the internal secret states of the participants in the traditional security model. However, in the actual application, a certain amount of leakage on the secret key can be captured by an adversary through performing various leakage attacks, such as a side-channel attack, a cold-boot attack. Thus, the previous cryptography schemes proved in the ideal security model cannot keep their claimed security in the leakage setting. The computational efficiency of the previous leakage-resilient certificateless key encapsulation mechanism (CL-KEM) is low due to a large amount of bilinear mapping. To improve the shortcoming, in this paper, a new leakage-resilient CL-KEM is proposed without using bilinear mapping, and the security of the proposed scheme is proved based on the classic decisional Diffie-Hellman assumption. In addition, all elements in the encapsulated ciphertext are random from the viewpoint of the adversary, which can make sure that any adversary cannot learn the leakage about the user's secret key from the corresponding given encapsulated ciphertext, and the leakage parameters are fixed constants, which is not limited by the size of encapsulation key space. In order to further enhance the leakage resilience, this paper constructs a novel leakage-resilient CL-KEM, in which the length of leakage can achieve $l_{\mathrm{sk}}(1-\mathcal{O}(1))$, where l_{sk} denotes the length of secret key. This scheme has the above advantages and the ability to resist leakage attack. In addition, we propose the generic constructions of leakage-resilient certificateless hybrid encryption and the leakage-resilient certificateless authenticated key exchange protocol based on the leakage-resilient CL-KEM.

Keywords certificateless public-key cryptography, key-encapsulation mechanism, leakage resilience, continuous leakage resilience, DDH security assumption

Yanwei ZHOU was born in 1986, and received his Ph.D. degree in computer software and theory from Shaanxi Normal University, in 2018. He is currently a senior engineer at the School of Computer Science, Shaanxi Normal University, China. His research interests include leakage-resilient cryptography and anonymous communication.



Zirui QIAO was born in 1985, and received her M.S. degree in computer software and theory from Shaanxi Normal University, in 2011. She is currently working toward her Ph.D. degree at the School of Computer Science of Shaanxi Normal University. Her main research interests include cryptography and information security.

Bo YANG was born in 1963, and received his Ph.D. degree in cryptography from Xidian University, in 1999. He is currently a professor at the School of Computer Science, Shaanxi Normal University. His research interests include information security and cryptography.

Zhe XIA was born in 1982, and received his Ph.D. degree from the University of Surrey, in 2009. He is currently an associate professor at the Department of Computer Science, Wuhan University of Technology, China. His research interests include secure e-voting protocols, secret sharing, and secure multiparty computation.