（5）《Fast and Secure Updatable Encryption》

时间：2022

会议：CRYPTO（A 会）

所看版本为 2022 年 ePrint 的第 4 版，为 CRYPTO 论文的扩充版。

主要内容：

1.概述

本文针对可更新加密（UE）的问题，针对之前文献，提出了一个更加安全的模型 xxIND-UE-atk，并基于此模型构造了 SHINE 方案，其存在 3 个变体，分别适用于短消息及长消息。

2.背景知识 updatable encryption （UE）

最原始的思想是 Key Rotation，周期性地对密文进行解密后再重新加密。要么所有者必须下载、重新加密并上传所有密文(这使得外包变得不切实际)，要么加密密钥必须发送给主机，这就违反了安全性。

Boneh【1】等人为此提出第一个 UE 方案，Client 生成一个 token 并将其发送给 Server，并允许 Server 在不对密文进行解密的情况下使用新的密钥更新密文。UE 方案其顺序地从一个周期（epoch）重新加密数据为下一周期的密文，并有如下特点：

- *周期性地更换密钥*
- *无需解密，直接更新密文*
- *通过 Update token 完成*

随后，又有许多学者对 UE 进行进一步研究，早期的研究中一个重要的区分就是方案是**密文独立（ciphertext-dependent）**的，即 token（尤其是 token 的 size）是否依赖于更新的密文（尤其是密文的数量）。并且，在后续的研究中，密文独立是一个研究趋势，近期的研究如【3】和【4】。

具体而言，【1】提出了一种密钥同态（key-homomorphic）PRF 的结构方案 BLMR，它本质上是一个对称代理重加密方案(proxy re-encryption scheme)，即每个周期更新密钥时对明文进行重加密。该方案由于以明文的形式附上了一个随机数（nonce），无法达到 IND-UPD 的安全性。

【3】是当前最贴近研究目标的一篇文章。该方案基于 symmetric-Elgamal 方案，被称为 RISE，其使用随机的 Update 算法并被证明在 DDH 假设下满足 IND-ENC 和 randIND-UPD 安全。其还通过在 BLMR 中加密随机数（nonce）将其优化为 BLMR+方案，并证明其在较弱的情况下是 IND-UPD 安全的。

【4】的目标是实现更强的安全性，即 CCA 安全+完整性保护。其指出【3】的 RISE 方案未能实现密文的完整性保护，他们的方案，基于 encrypt-and-MAC，但是相比 RISE 方案牺牲了一定效率。

本文的目标是实现【4】的 CCA 安全+完整性，并且实现和【3】相当的效率。

3.本文贡献

- *给出了更强的安全性定义，其加密算法与更新算法输出的密文不可区分。**不仅提供了之前文献期望的不可链接性**，也回答了 KLR19 文献提到的开放性问题；*
- *设计一个高效的 SHINE 算法，介绍了其 3 个变体 SHINE0，MirrorSHINE 和 OCBSHINE，并且就安全性、密文拓展性及效率与前人文献进行了对比，如图：*

	Security		Ciphertext Expansion		Efficiency
	IND	INT	$\|M\|$	$\|C\|$	Enc (Upd)
BLMR [BLMR13]	(det, ENC, CPA)	✘	$l\|\mathbb{G}\|$	$(l+1)\|\mathbb{G}\|$	$l\mathbf{E}$
BLMR+ [BLMR13, LT18a]	(weak, UE, CPA)	✘	$l\|\mathbb{G}\|$	$(l+1)\|\mathbb{G}\|$	$l\mathbf{E}$
RISE [LT18a]	(rand, UE, CPA)	✘	$1\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$2\mathbf{E}$
SHINE0[CPA] § 5.1.1	(det, UE, CPA)	✘	$(1-\gamma)\|\mathbb{G}\|$	$1\|\mathbb{G}\|$	$1\mathbf{E}$
NYUE [KLR19a]	(rand, ENC, RCCA) (rand, UPD, RCCA)	✘	$1\|\mathbb{G}_1\|$	$(34\|\mathbb{G}_1\|, 34\|\mathbb{G}_2\|)$	$(60\mathbf{E}, 70\mathbf{E})$
NYUAE [KLR19a]	(rand, ENC, RCCA) (rand, UPD, RCCA)	PTXT	$1\|\mathbb{G}_1\|$	$(58\|\mathbb{G}_1\|, 44\|\mathbb{G}_2\|)$	$(110\mathbf{E}, 90\mathbf{E})$
E&M [KLR19a]	(det, ENC, CCA) (det, UPD, CCA)	CTXT	$1\|\mathbb{G}\|$	$3\|\mathbb{G}\|$	$3\mathbf{E}$
SHINE0 § 5.1.1	(det, UE, CCA)	CTXT	$(1-2\gamma)\|\mathbb{G}\|$	$1\|\mathbb{G}\|$	$1\mathbf{E}$
MirrorSHINE § 5.1.2	(det, UE, CCA)	CTXT	$(1-\gamma)\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$2\mathbf{E}$
OCBSHINE § 5.1.3	(det, UE, CCA)	CTXT	$l\|\mathbb{G}\|$	$(l+2)\|\mathbb{G}\|$	$(l+2)\mathbf{E}$

4.UE 下的安全模型

对于 UE 方案的安全性而言，一个开放的问题：是随着更多特性被考虑用来抵抗敌手的攻击，目前还不明确哪些安全目标是必要且最优的。

以前的安全定义不能保证攻击者无法区分当前周期的新密文 vs 上一周期的旧密文，因此本文给出了更强的安全定义。此外，本文还结合【5】考虑了密文的完整性。

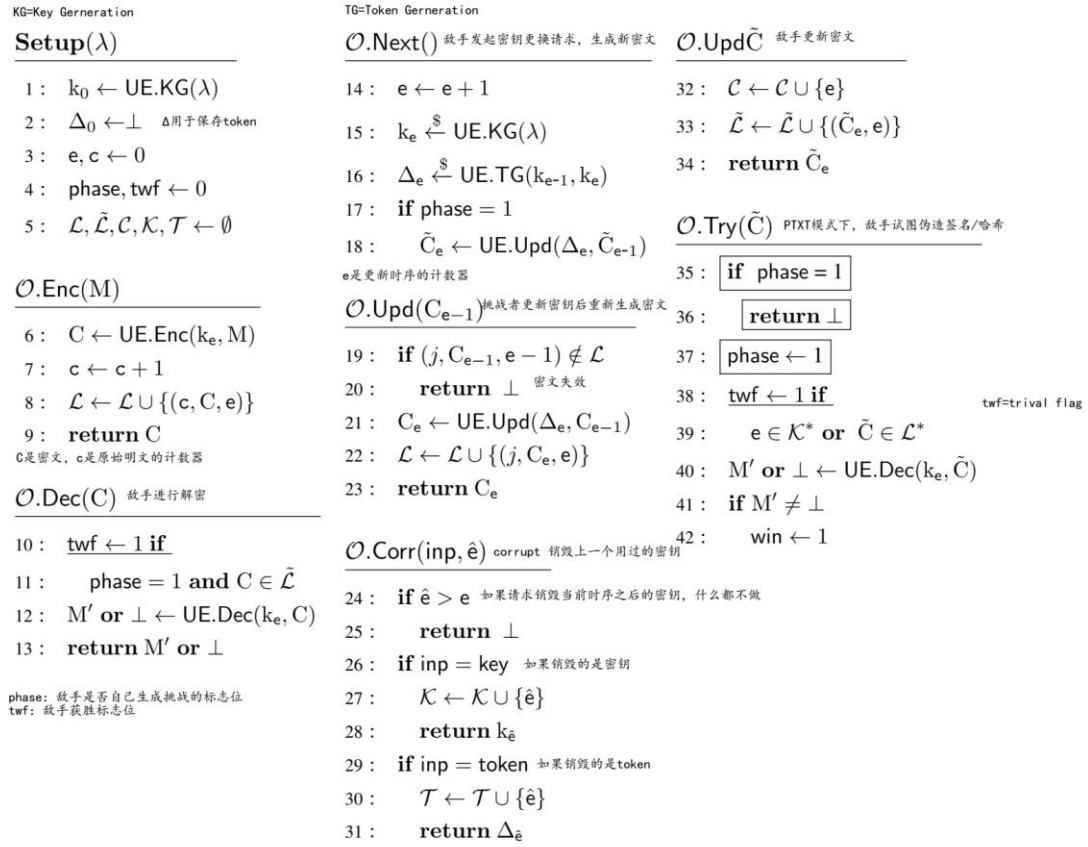
对于 xxIND-yy-atk-b，xx、yy、b 的所有可能取值如表所示。

名称/编号	**1**	**2**	**3**
xx	det(确定性)	rand（随机性）	
yy	ENC	UPD	UE

atk	CPA	CCA	

1） UE 下的应答器定义

根据【4】的定义，对于保密性的敌手-挑战者游戏定义 oracle 如下：



以下是符号的描述

- $\mathcal{L}$: List of non-challenge ciphertexts (from $\mathcal{O}$.Enc or $\mathcal{O}$.Upd) with entries of form (c, C, e), where query identifier c is a counter incremented with each new $\mathcal{O}$.Enc query. 非挑战密文（不是敌手主动询问应答器的信息）
- $\tilde{\mathcal{L}}$: List of updated versions of challenge ciphertext (created via $\mathcal{O}$.Next, received by adversary via $\mathcal{O}$.Upd$\tilde{\mathrm{C}}$), with entries of form $(\tilde{C}, e)$. 挑战密文（敌手主动询问应答器的信息）

Further, we use the following lists that track epochs only.

- $\mathcal{C}$: List of epochs in which adversary learned updated version of challenge ciphertext (via CHALL or $\mathcal{O}$.Upd$\tilde{\mathrm{C}}$). 密文的集合
- $\mathcal{K}$: List of epochs in which the adversary corrupted the encryption key. 不同时期敌手破坏密钥的集合
- $\mathcal{T}$: List of epochs in which the adversary corrupted the update token. 不同时期敌手破坏Token的集合

备注：名词解释（Post-Compromise Security）后向妥协安全是指过期的密钥即使被重新利用，也仍然是安全的。

1） 胜利条件

作者在文章中介绍了 3 种敌手攻破保密性的条件以及 2 种攻破完整性的条件

保密性（3 种）

- **Trivial wins via keys and ciphertexts.**
- **Trivial wins via direct updates.**
- **Trivial wins via decryptions.**

完整性（2 种）

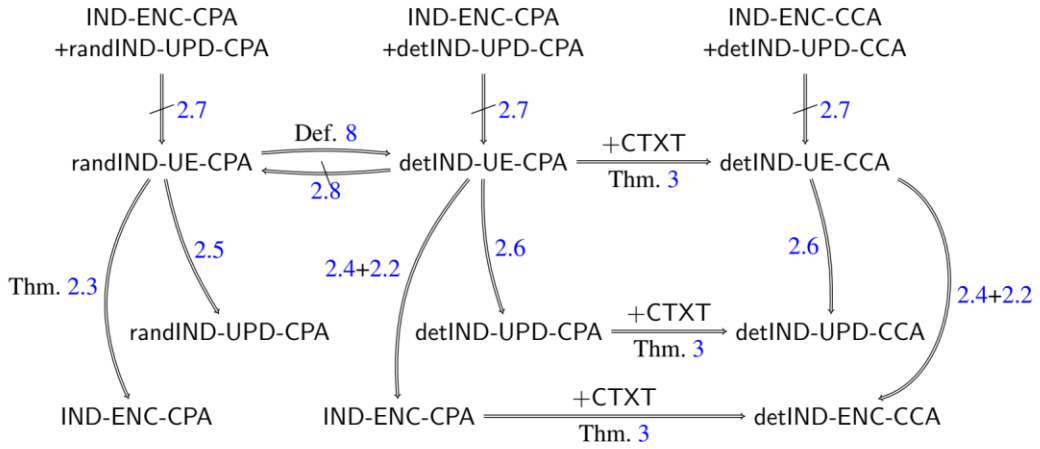
- **Trivial wins via keys.**
- **Trivial wins via ciphertexts.**

Notion	$\mathcal{O}$.Enc	$\mathcal{O}$.Dec	$\mathcal{O}$.Next	$\mathcal{O}$.Upd	$\mathcal{O}$.Corr	$\mathcal{O}$.Upd$\tilde{\mathrm{C}}$	$\mathcal{O}$.Try
detIND-yy-CPA	✓	×	✓	✓	✓	✓	×
randIND-yy-CPA	✓	×	✓	✓	✓	✓	×
detIND-yy-CCA	✓	✓	✓	✓	✓	✓	×
INT-CTXT	✓	×	✓	✓	✓	×	✓

Figure 7: Oracles the adversary is allowed to query in different security games, where yy ∈ {ENC, UPD, UE}. × indicates the adversary does not have access to the corresponding oracle,✓ indicates the adversary has access to the corresponding oracle. 每个安全游戏中所访问的应答器

2） 安全性的关系及证明

Theorem 2 (Informal Theorem)**.** The relationship among the security notions xxIND-UE-atk, xxIND-ENC-atk and xxIND-UPD-atk are as in Fig. 18. This is proven via Theorem 2.2 to 2.8, and Theorem 3.



5. 本文的 3 种 SHINE 方案

作者基于更严格的 UE 安全，提出了 **SHINE** 方案（Secure Homomorphic Ideal-cipher Nonce-based Encryption），其主要包含如下 3 个版本。

方案一 SHINE0：用于短消息，只执行 1 次置换

KG（Key Gen）：随机生成 1 个密钥

TG（Token Gen）：计算下一周期的更新 token Δe+1，值为 2 个密钥的商。

Enc :计算密文，生成一个随机数（nonce），连接新鲜数+消息+0block 后先计算置换再求

幂

Dec：解密，先开方后再计算置换的逆；

Upd：用新的密钥加密密文，对应的操作就是计算原密文的 token 次幂。

备注：0 block 用于保护完整性，完整性保护是可选项。其完整保护主要是通过在消息后面附加 1 串 0，并对其进行完整性保护，若验证时仍为一串 0，说明消息没有被篡改过。

SHINE0.KG(λ)

1： $\mathrm{k} \xleftarrow{\$} \mathbb{Z}_q^*$

2： **return** k

SHINE0.TG($\mathrm{k_e}, \mathrm{k_{e+1}}$)

3： $\Delta_{e+1} \leftarrow \frac{\mathrm{k_{e+1}}}{\mathrm{k_e}}$

4： **return** Δ_{e+1}

SHINE0.Enc($\mathrm{k_e}, \mathrm{M}$)

5： $\mathrm{N} \xleftarrow{\$} \mathcal{N}$

6： $\mathrm{C_e} \leftarrow (\pi(\mathrm{N}\|\mathrm{M}\|0^{\mathrm{t}}))^{\mathrm{k_e}}$

7： **return** $\mathrm{C_e}$

SHINE0.Dec($\mathrm{k_e}, \mathrm{C_e}$)

8： $a \leftarrow \pi^{-1}(\mathrm{C_e}^{1/\mathrm{k_e}})$

9： **parse**† a **as** $\mathrm{N}'\|\mathrm{M}'\|\mathrm{Z}$

10： **if** $\mathrm{Z} = 0^{\mathrm{t}}$

11： **return** M'

12： **else**

13： **return** $\perp$

SHINE0.Upd($\Delta_{e+1}, \mathrm{C_e}$)

14： $\mathrm{C_{e+1}} \leftarrow (\mathrm{C_e})^{\Delta_{e+1}}$

15： **return** $\mathrm{C_{e+1}}$

方案二 irrorSHINE：执行 2 个不同的置换

较方案一适用与更长消息，其主要区别是完整性通过 2 个不同的随机置换对同一消息来实现。

备注：其完整保护主要是通过 2 个不同的随机置换算法生成校验值，若反向计算后 2 个值相同，说明消息没有被篡改过。

MirrorSHINE.KG(λ)

1： $\mathrm{k} \xleftarrow{\$} \mathbb{Z}_q^*$

2： **return** k

MirrorSHINE.TG($\mathrm{k_e}, \mathrm{k_{e+1}}$)

3： $\Delta_{e+1} \leftarrow \frac{\mathrm{k_{e+1}}}{\mathrm{k_e}}$

4： **return** Δ_{e+1}

MirrorSHINE.Enc($\mathrm{k_e}, \mathrm{M}$)

5： $\mathrm{N} \xleftarrow{\$} \mathcal{N}$

6： $\mathrm{C_e^1} \leftarrow (\pi_1(\mathrm{N}||\mathrm{M}))^{\mathrm{k_e}}$

7： $\mathrm{C_e^2} \leftarrow (\pi_2(\mathrm{N}||\mathrm{M}))^{\mathrm{k_e}}$

8： $\mathrm{C_e} \leftarrow (\mathrm{C_e^1}, \mathrm{C_e^2})$

9： **return** $\mathrm{C_e}$

MirrorSHINE.Dec($\mathrm{k_e}, \mathrm{C_e}$)

10： **parse** $\mathrm{C_e} = (\mathrm{C_e^1}, \mathrm{C_e^2})$

11： $a_1 \leftarrow \pi_1^{-1}((\mathrm{C_e^1})^{1/\mathrm{k_e}})$

12： $a_2 \leftarrow \pi_2^{-1}((\mathrm{C_e^2})^{1/\mathrm{k_e}})$

13： **parse**‡ a_1 **as** $\mathrm{N}'||\mathrm{M}'$

14： **if** $a_1 = a_2$

15： **return** M'

16： **else**

17： **return** $\perp$

MirrorSHINE.Upd($\Delta_{e+1}, \mathrm{C_e}$)

18： **parse** $\mathrm{C_e} = (\mathrm{C_e^1}, \mathrm{C_e^2})$

19： $\mathrm{C_{e+1}^1} \leftarrow (\mathrm{C_e^1})^{\Delta_{e+1}}$

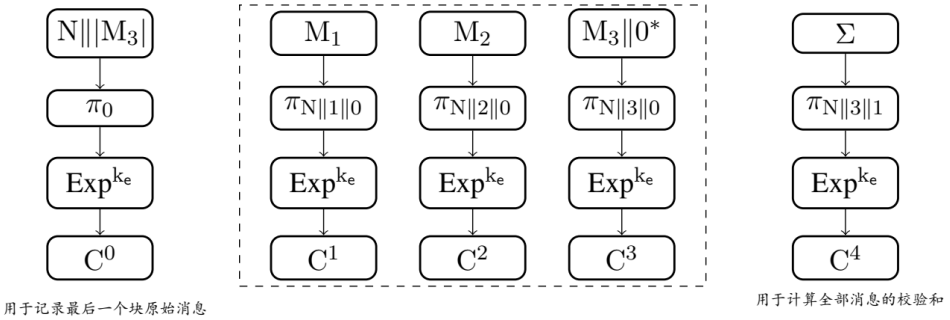
20： $\mathrm{C_{e+1}^2} \leftarrow (\mathrm{C_e^2})^{\Delta_{e+1}}$

21： **return** $\mathrm{C_{e+1}^1}, \mathrm{C_{e+1}^2}$

方案三：OCBSHINE：适用于任意长的消息，执行多个置换（permutation family）

由于方案一和方案二在原理上要求消息的空间小于生成元求幂构成群的空间，因此不适用于任意长的消息。该方法借鉴了 OCB 认证加密模式【2】的思路。

OCBSHINE 方案将 m 长的消息进行分组，分为 l 个消息块 M_1; ...; M_l，每个分组的长度是 m，M_1, ..., M_l 用于保存实际消息并加密为 C_1,…,C_l（最后一个分组 M_l 若不足长度则填充 0）。同时，其维护 2 个额外的数据块，C_0 用于记录最后一个块原始消息，C_{l+1} 用于记录全部消息的校验和，如图。



具体算法如下图：

$\mathsf{OCBSHINE.KG}(\lambda)$

1 : $\mathrm{k} \xleftarrow{\$} \mathbb{Z}_q^*$
2 : **return** k

$\mathsf{OCBSHINE.TG}(\mathrm{k_e}, \mathrm{k_{e+1}})$

3 : $\Delta_{e+1} \leftarrow \dfrac{\mathrm{k_{e+1}}}{\mathrm{k_e}}$
4 : **return** Δ_{e+1}

$\mathsf{OCBSHINE.Enc}(\mathrm{k_e}, \mathrm{M})$

5 : partition M into $\mathrm{M}_1, \ldots, \mathrm{M}_l$
6 : $\Sigma \leftarrow \oplus_{i=1}^{l-1} \mathrm{M}_i \oplus \mathrm{M}_l \| 0^*$
随机生成m-a比特的随机数 7 : $\mathrm{N} \xleftarrow{\$} \mathcal{N}$
额外生成的2个数据块 8 : $\mathrm{C}^0 \leftarrow \left(\pi_0(\mathrm{N}||\mathrm{M}_l|)\right)^{\mathrm{k_e}}$
9 : $\mathrm{C}^{l+1} \leftarrow \left(\pi_{\mathrm{N}\|l\|1}(\Sigma)\right)^{\mathrm{k_e}}$
10 : **for** $i = 1, \ldots, l$-1 **do**
11 : $\mathrm{C}^i \leftarrow \left(\pi_{\mathrm{N}\|i\|0}(\mathrm{M}_i)\right)^{\mathrm{k_e}}$
填充最后一个消息块 12 : $\mathrm{C}^l \leftarrow \left(\pi_{\mathrm{N}\|l\|0}(\mathrm{M}_l\|0^*)\right)^{\mathrm{k_e}}$
13 : $\mathrm{C_e} \leftarrow (\mathrm{C}^0, \ldots, \mathrm{C}^l, \mathrm{C}^{l+1})$
14 : **return** $\mathrm{C_e}$

$\mathsf{OCBSHINE.Dec}(\mathrm{k_e}, \mathrm{C_e})$

15 : **parse** $\mathrm{C_e} = (\mathrm{C}^0, \ldots, \mathrm{C}^l, \mathrm{C}^{l+1})$
16 : $\mathrm{N}'\|A' \leftarrow \pi_0^{-1}\left((\mathrm{C}^0)^{1/\mathrm{k_e}}\right)$
17 : $\Sigma' \leftarrow \pi_{\mathrm{N}'\|l\|1}^{-1}\left((\mathrm{C}^{l+1})^{1/\mathrm{k_e}}\right)$ 还原出校验和
18 : **for** $i = 1, \ldots, l$ **do**
19 : $\mathrm{M}'_i \leftarrow \pi_{\mathrm{N}'\|i\|0}^{-1}\left((\mathrm{C}^i)^{1/\mathrm{k_e}}\right)$
20 : **if** $\Sigma' = \oplus_{i=1}^{l} \mathrm{M}'_i$ 计算实际校验和并比对
21 : $\mathrm{M}' \leftarrow (\mathrm{M}'_1, \ldots, \mathrm{M}'_l[x])$
22 : **return** M'
23 : **else**
24 : **return** $\perp$

$\mathsf{OCBSHINE.Upd}(\Delta_{e+1}, \mathrm{C_e})$

25 : **parse** $\mathrm{C_e} = (\mathrm{C}^0, \ldots, \mathrm{C}^l, \mathrm{C}^{l+1})$
26 : **for** $i = 0, \ldots, l+1$ **do**
27 : $\mathrm{C}^i_{e+1} \leftarrow (\mathrm{C}^i_e)^{\Delta_{e+1}}$
28 : **return** $\mathrm{C_{e+1}}$

作者逐步地证明了 SHINE 方案是同时满足 detIND-UE-CPA 安全及 INT-CTXT 安全的。

在实现阶段，作者给出了建议，对于随机置换，AES128 不能满足群的最小规模，建议

使用 Treefish、original Rijndael（256 bits or 512 bits）；对于求幂，建议使用椭圆曲线，如 NIST P-256 or P-512。并且 256bits 的 Rijndael 需要配合 NIST P-256，并且 512bits 的 Rijndael 需要配合 NIST P-512。

参考文献

【1】 Boneh, D., Lewi, K., Montgomery, H., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 410–428. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4 23

【2】 Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: a block-cipher mode of operation for efficient authenticated encryption. In Michael K. Reiter and Pierangela Samarati, editors, *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6-8, 2001*, pages 196–205. ACM, 2001.Cited on page 32.

【3】 Anja Lehmann and Björn Tackmann. Updatable encryption with post-compromise security. In Jesper Buus Nielsen and Vincent Rijmen, editors, Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III, volume 10822 of Lecture Notes in Computer Science, pages 685–716. Springer, 2018. Cited on pages 3, 4, 6, 7, 16, 51, and 56

【4】 Michael Klooß, Anja Lehmann, and Andy Rupp. (R)CCA secure updatable encryption with integrity protection. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 68–99. Springer, 2019. Cited on pages 3, 5, 7, 9, 16, 19, and 20.

【5】 Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptology*, 21(4):469–491, 2008. Cited on page 4