Hindawi
Security and Communication Networks
Volume 2019, Article ID 8357241, 8 pages
https://doi.org/10.1155/2019/8357241

WILEY | Hindawi

Research Article

CCA Secure Public Key Encryption against After-the-Fact Leakage without NIZK Proofs

Yi Zhao,[1,2] **Kaitai Liang**,[3] **Bo Yang**,[1,2] **and Liqun Chen**[3]

[1]*Computer Science, Shaanxi Normal University, 710119 West Changan Street 620, Xi'an, Shaanxi, China*
[2]*State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China*
[3]*Computer Science, University of Surrey, GU27XH Guildford, UK*

Correspondence should be addressed to Bo Yang; byang@snnu.edu.cn

Received 10 August 2018; Accepted 3 September 2019; Published 31 October 2019

Academic Editor: Francesco Gringoli

In leakage resilient cryptography, there is a seemingly inherent restraint on the ability of the adversary that it cannot get access to the leakage oracle after the challenge. Recently, a series of works made a breakthrough to consider a postchallenge leakage. They presented achievable public key encryption (PKE) schemes which are semantically secure against after-the-fact leakage in the split-state model. This model puts a more acceptable constraint on adversary's ability that the adversary cannot query the leakage of secret states as a whole but the functions of several parts separately instead of prechallenge query only. To obtain security against chosen ciphertext attack (CCA) for PKE schemes against after-the-fact leakage attack (AFL), existing works followed the paradigm of "double encryption" which needs noninteractive zero knowledge (NIZK) proofs in the encryption algorithm. We present an alternative way to achieve AFL-CCA security via lossy trapdoor functions (LTFs) without NIZK proofs. First, we formalize the definition of LTFs secure against AFL (AFLR-LTFs) and all-but-one variants (ABO). Then, we show how to realize this primitive in the split-state model. This primitive can be used to construct AFLR-CCA secure PKE scheme in the same way as the method of "CCA from LTFs" in traditional sense.

1. Introduction

In the past two decades, physical attacks which are capable of getting access to partial information of the secret state have become a serious threat to the security of cryptographic algorithms in practice. These attacks have moved far beyond the scope of traditional cryptography with an inherent assumption that no information of the secret key is leaked. Up till now, the branch of cryptography to treat this issue is highly motivated, which is called leakage resilient cryptography.

The first step to address leakage resilience systematically is formalizing the leakage attack in the traditional security notion. There are already several models in the existing works which describe leakage in different ways. Akavia et al. [1] modeled the leakage as the bounded output of an arbitrary function of secret states (bounded/relative leakage). Naor and Segev [2] presented an alternative description to allow leakage without length restriction. They measured the leakage by the induced decrease of the minimum entropy of the secret (noisy leakage). Under these formulations, some leakage resilient primitives are successfully designed, including signature schemes [3–5] and key agreement protocol [6, 7].

However, in the area of public key encryption (PKE), there is an inherent restriction in the security notion. Semantic security is always defined to be the indistinguishability of the challenge ciphertext issued by an adversary in a game with a challenger answering different types of queries from the adversary. The full-fledged definition for leakage resilience allows the adversary to query the leakage oracle after the challenge. This means an adversary could design its leakage function via the information of challenge ciphertext. For instance, in the bounded leakage model, an adversary could encode the challenge and decryption algorithm together to recover the

whole message via leakage queries if its length is shorter than the bound. Most existing works, such as [1, 2, 8–10], beg this technical difficulty with a weaker security definition, which only admits prechallenge leakage queries. But in practice, after-the-fact leakage is really feasible because many cryptographic devices are portable so that the attack can be launched at any time.

Halevi and Lin [11] made an effort to treat after-the-fact leakage (AFL) directly. As classic semantic security is impossible under postleakage attack, they choose to put another limitation instead of ignoring it. They require that the adversary can only get access to different parts of the secret via leakage independently, not as a whole. This "split-state" leakage was also defined and applied in the setting "computation leaks only" [12]. This restriction is meaningful because it is feasible to store secret fractions in different locations. They introduced the notion of "entropy leakage" to capture after-the-fact leakage. This concept states that the leakage should not be used to obtain more information than itself. This is an essential property for a postchallenge leakage. They showed that constructions from the hash proof system like that in [2] meet the requirement of security against entropy leakage. And they gave the first after-the-fact leakage resilient (AFLR) encryption scheme secure against chosen plaintext attack (CPA) by combining two instances of entropy leakage resilient schemes. Then, Li et al. presented identity-based encryption secure against postchallenge leakage attack [13]. Yang and Li considered this problem for the key exchange protocol [14].

Since security against chosen ciphertext attack (CCA) is a well-accepted standard for encryption schemes, some subsequent works aimed to achieve this goal against AFL. Zhang et al. [15] followed the classic Naor–Yung paradigm [16] to give a construction with simulation sound noninteractive zero knowledge (NIZK) proof. Chakraborty et al. [17] presented a more efficient construction with true simulation extractable NIZK proof. Fujisaki et al. [18] considered the multichallenge setting as well as the leakage from randomness. There are indeed more techniques to obtain traditional CCA security, but few existing works secure against AFL attacks have been proposed.

Lossy trapdoor functions: besides double encryption paradigm [16] and hash proof system [19], there is another approach to achieve CCA security, via a powerful primitive called lossy trapdoor functions (LTFs). Since its appearance [20], this primitive has been widely applied in many areas. The CCA secure encryption schemes based on LTFs get rid of the burden from NIZK proofs so that it is more efficient than those which need NIZK proofs. Also, LTFs have brilliant properties to extract statistical entropy from computational indistinguishability between two working modes. So LTFs have its nature to play an important role in leakage resilient cryptography. Some prior works already tried this way. Qin et al. [8] designed an invariant called the lossy filter to replace the universal-2 part in HPS-based schemes and achieved better leakage rate. More directly, Qin et al. [21] attempted to construct LR-LTFs, but their result can only be proven secure in a weaker model in which the adversary can get access to entire public key after leakage queries. Chen et al. proposed an advanced version of lossy function with its application in leakage resilience [22].

1.1. Our Contribution. In this work, we demonstrate that AFLR-LTFs and ABO invariants can be constructed in the split-state model and then can achieve AFLR-CCA security without NIZK proofs either. First, we formulate the notion of AFL secure LTFs. Then, we realize this primitive from AFL CPA secure PKE schemes. To overcome the technical difficulty that most randomness extractors and the underlying PKE schemes do not have homomorphic property which is essential for this use, we refine a AFLR randomness extractor from the BHHO PKE scheme [21, 23] with this property. Thus, with an AFLR-LTF and an AFL-ABO-LTF, we can follow the approach in [20] to achieve CCA security. Furthermore, our construction is easy to be used to construct chameleon AFL-ABO-LTFs [24] for a more efficient CCA secure realization.

1.2. Organization. The remaining part of this paper is organized as follows: the basic definitions and tools we need is shown in Section 2. In Section 3, we build a step stone before arriving to the final step: a two-source extractor in the 2 split-state model. Then, we present AFLR-LTFs in Section 4 and an AFLR PKE scheme based on them in Section 5, respectively. The final scheme is interpreted in a black box manner from AFLR-LTFs. The security of the final scheme can be reduced to the security of AFLR-LTFs.

2. Preliminaries

2.1. ABO Lossy Trapdoor Functions. A collection of LTFs is a collection of publicly computable functions which are indexed by a set of public key $\{s\}$. Every public key is associated with a branch which is used to generate the key. There are two kinds of public keys. Functions indexed by one kind are injective, while functions indexed by the other have a smaller size of image than that of domain. We called the branch according to the former "injective branch" and the other "lossy branch." "Lossy" means the image of the function working on these branches loses part of the information of the preimage. We use a generalized notion to incorporate exponential lossy branches. Let $\{B_n\}$ denote a collection of branch sets and $\{B_n^*\}$ denote the corresponding collection of lossy branch sets. We recall the definition of ABO-LTFs [20] below. If $\{B_n\}$ contains two elements only, it is just the standard LTF.

Definition 1. A collection of (n, k) ABO-LTFs is composed of 3 probabilistic polynomial time (PPT) algorithms:

G_{abo}: take $\lambda \in \mathbb{N}$ and $b^* \in B_\lambda$ as input and output $(s, \text{td}, B_\lambda^*)$, where s is a function index, td is its trapdoor, and B_λ^* is the set of lossy branches that $b^* \in B_\lambda^*$.

F_{abo} and F_{abo}^{-1}: for any $b \in B_\lambda / B_\lambda^*$, $F_{\text{abo}}(s, b, \cdot)$ computes an injective function $f_{s,b}(\cdot)$ over the domain $\{0,1\}^n$ and $F_{\text{abo}}^{-1}(s, b, \cdot)$ computes $f_{s,b}^{-1}(\cdot)$. For any $b \in B_\lambda^*$, $F_{\text{abo}}(s, b, \cdot)$

computes a function $f_{s,b}(\cdot)$ over the domain $\{0,1\}^n$ whose image size is at most 2^k.

There are two security requirements for ABO-LTFs. Index indistinguishability: the ensemble $s \longleftarrow G_{\text{abo}}(\lambda, b_0^*)$ and $s \longleftarrow G_{\text{abo}}(\lambda, b_1^*)$ are computationally indistinguishable. Lossy branch hidden: any PPT adversary $\mathcal{A}$ which takes (s, b) as input, where $(s, \text{td}, B^*) \longleftarrow G_{\text{abo}}(\lambda, b^*)$ has only a negligible probability to find a b' such that $b' \neq b$ and $b' \in B^*$. And even $b \in B^*$, and the adversary could not find one either.

2.2. Randomness Extractor

2.2.1. One Source

Definition 2. A randomized algorithm $\text{Ext1}: \mathcal{K} \longrightarrow \{0,1\}^\nu$ is a (μ, ε) extractor if for all (X, Z) that is distributed on $\mathcal{K}$ and $\tilde{H}_\infty(X \mid Z) \geq \mu$, $(Z, S, \text{Ext1}(X; S)) =_s (Z, S, U_\nu)$, where U_ν is a uniform distribution over $\{0,1\}^\nu$ and S is called seed which is the coin of Ext1.

The parameters of the concrete extractor used need to satisfy the condition that $\nu \leq \mu - 2\log(1/\varepsilon) - 1$. Generally, pair-wise independent hash functions are used to realize extractors.

2.2.2. Two Source

Definition 3. A two-source extractor does not rely on random seeds but extracts randomness from two independent sources. A randomized algorithm $\text{Ext2}: (\mathcal{K})^2 \longrightarrow \{0,1\}^\nu$ is a (μ, ε) extractor if for all (K_1, K_2, Z) where K_1 and K_2 are distributed on $\mathcal{K}$ and have minimum entropy μ conditioned on Z, $(Z, \text{Ext}(K_1, K_2)) =_s (Z, U_\nu)$.

2.3. AFLR-CPA Secure PKE

2.3.1. Entropy Leakage Resilient PKE. The definition of entropy leakage resilience stresses that the leakage after challenge cannot be amplified. This fact is captured by a simulator, which interacts with the adversary in an indistinguishable manner to the real setting. Formally, we first set some parameters: k is the minimum entropy that the message source M has, l_{post} denotes the leakage after challenge, and δ is an overhead parameter which comes from the statistical distance that the extractor deviates from uniform distribution.

Definition 4. A PKE scheme Π is entropy leakage resilient if there exists a simulator Sim such that, for every PPT adversary $\mathcal{A}$, the following two conditions hold:

(1) $(M^{\text{real}}, \text{View}_{\mathcal{A}}^{\Pi}) =_c (M^{\text{sim}}, \text{View}_{\mathcal{A}}^{\text{Sim}})$

(2) $\tilde{H}_\infty(M^{\text{sim}} \mid \text{View}_{\mathcal{A}}^{\text{Sim}}) \geq k - l_{\text{post}} - \delta$

2.3.2. After-the-Fact Leakage Resilience. Semantic security against AFL is defined by a game between a challenger and an adversary just the same as normal CPA game, except that the adversary is allowed to issue leakage query before and after challenge. The semantic security requires that the adversary can still not win with nonnegligible advantage in this setting. The CCA security is define analogously.

An AFLR-CPA-secured PKE scheme in the 2 split-state model can be constructed via combination of two instances of an entropy leakage resilient PKE scheme and a two-source extractor. Specifically, given two entropy leakage resilient PKE schemes $\Pi_1 = (\text{Gen}_1, \text{Enc}_1, \text{Dec}_1)$ and $\Pi_2 = (\text{Gen}_2, \text{Enc}_2, \text{Dec}_2)$, a semantic secure scheme against a posthallenge leakage can be defined as $\text{Gen} = (\text{Gen}_1, \text{Gen}_2)$, $\text{Enc}(m; r_1, r_2) = (c_1 = \text{Enc}_1(r_1), c_2 = \text{Enc}_2(r_2), \text{Ext2}(r_1, r_2) \oplus m) = c$, $\text{Dec}(c) = c \oplus \text{Ext2}(\text{Dec}_1(c_1), \text{Dec}_2(c_2))$. The security proof for this construction is in [11].

2.4. Homomorphism.

A function is called homomorphism if the operation between elements in the domain preserves its structured functionality between elements in the range. For instance, let "+" denote the operation in the domain, "·" denote the operation in the range, and f: $A \longrightarrow B$ be the function. The property can be represented as $f(x + y) = f(x) \cdot f(y)$, which can induce $f(ax) = f(x)^a$.

2.5. DDH Assumption.

Given a cyclic group G with order q which is a big prime number, f and h are random elements in G and then (f, h, f^r, h^r) and $(f, h, f^r, h^{r'})$ are computationally indistinguishable for randomly chosen r and r'. Following a hybrid argument, this result can be extended to vector situation: $(g, f_1^r, \ldots, f_l^r, h_1^r, \ldots, h_l^r)$ and $(g, f_1^r, \ldots, f_l^r, h_1^{r'}, \ldots, h_l^{r'})$ are computationally indistinguishable for randomly chosen r and r'.

2.6. 2 Split-State Model.

This model is introduced in [11] to incorporate postchallenge leakage resilience. This model puts one more restriction than the ordinary security model against leakage attack that an adversary cannot issue leakage queries on the whole secret state but two separate parts. This means, instead of a leakage function f on sk, the adversary can only issue queries f_1 on sk_1 and f_2 on sk_2.

2.7. Notations.

Throughout this paper, we build our concrete construction on quadratic residue subgroup of the cyclic group with order N^2. So we present all the parameter settings here. Let $\mathbb{G}$ denote a group of order N^2 where N is a Blum integer, $\mathbb{G}_r$ the subgroup of $\mathbb{G}^*$ with order $(p-1)(q-1)/4$, n the security parameter, λ the length of leakage, and set $l = 2 + (\lambda + 2(\log 1/\varepsilon)/\log N - 3)$ for some negligible ε.

Note that DDH assumption also holds in $\mathbb{G}_r$.

Also, we define the multiple computation and exponential computation of a vector as $xy = (x_1, \ldots, x_n)(y_1, \ldots, y_n) = (x_1 y_1, \ldots, x_n y_n)$ and $x^r = (x_1, \ldots, x_n)^r = (x_1^r, \ldots, x_n^r)$.

3. Homomorphic and Leakage Resilient Randomness Extraction

In general, the keyed randomness extractor in leakage resilient setting is initiated with universal hash functions which do not incorporate homomorphic property. However, it is quite vital in our scheme. So we refine an extractor from a variant of BHHO scheme [21] which meets our requirement and leads to a construction of homomorphic two-source extractor.

3.1. One-Source Leakage Resilient Extractor. An extractor can be constructed as follows with abovementioned parameters:

Gen: choose $f_1, f_2, \ldots, f_l \in G_r$. The evaluation key is set to be $\mathrm{pk} = (N, f_1, f_2, \ldots, f_l)$.

Ext: for any $x = (x_1, x_2, \ldots, x_l)$ sampled from source $\mathbb{Z}_{N-1/4}$, choose $r \in \mathbb{Z}_{N-1/4}$, compute public random seed $u_1 = f_1^r, u_2 = f_2^r, \ldots, u_l = f_l^r$, and $f = \prod_{i=1}^{l} f_i^{x_i}$. Then, the extracted randomness is $\mathrm{Ext}(x; r) = \prod_{i=1}^{l} u_i^{x_i}$.

Following [21], R is distributed negligibly close to uniform even subject to λ bits leakage and published f.

Homomorphic property: we observe that $\mathrm{Ext}(x; r)^a = \mathrm{Ext}(x; ar)$. So this extractor has homomorphic random seed.

3.2. Two Sources in 2 Split-State Model. Given the same parameters as above, we can present our publicly computable two-source extractor in 2 split-state model.

Gen: choose $f_1, f_2, \ldots, f_l, h_1, h_2, \ldots, h_l \in G_r$. The evaluation key is set to be $\mathrm{pk} = (N, f_1, f_2, \ldots, f_l, h_1, h_2, \ldots, h_l)$.

Ext2: for any $x = (x_1, x_2, \ldots, x_l)$ and $y = (y_1, y_2, \ldots, y_l)$ sampled from source $\mathbb{Z}_{N-1/4}$, choose $r \in \mathbb{Z}_{N-1/4}$, compute public random seed $u_1 = f_1^r, u_2 = f_2^r, \ldots, u_l = f_l^r$, $v_1 = h_1^r, v_2 = h_2^r, \ldots, v_l = h_l^r$, and $f = \prod_{i=1}^{l} f_i^{x_i}$ and $h = \prod_{i=1}^{l} h_i^{y_i}$. Then, the extracted randomness is $\mathrm{Ext2}(x, y) = \prod_{i=1}^{l} u_i^{x_i} v_i^{y_i}$.

Theorem 1. *The construction above is a* $(\log N - 3 - \lambda, \varepsilon)$ *two-source extractor against* λ *bits leakage under DDH assumption.*

Proof. We prove this theorem via hybrid argument through games between a challenger and an adversary as follows:

Game0: the game proceeds as the real game. The challenger chooses $\mathrm{pk} = (N, f_1, f_2, \ldots, f_l, h_1, h_2, \ldots, h_l)$ and responds queries from the adversary as the algorithm.

Game1: in this game, the only change is that challenger computes the public random seed with two randomness r and r', which is $u_1 = f_1^r, u_2 = f_2^r, \ldots, u_l = f_l^r$, $v_1 = h_1^{r'}, v_2 = h_2^{r'}, \ldots, v_l = h_l^{r'}$ and $f = \prod_{i=1}^{l} f_i^{x_i}$ and $h = \prod_{i=1}^{l} h_i^{y_i}$. □

Lemma 1. *The view of adversary is indistinguishable between Game0 and Game1 assuming DDH problem is hard.*

Given a DDH instance $(g_1, g_2, A = g_1^r, z)$, the challenger can simulate the game by letting $(f_1 = g_1^{x_1}, f_2 = g_1^{x_2}, \ldots, f_l = g_1^{x_l})$ and $(h_1 = g_2^{y_1}, h_2 = g_2^{y_2}, \ldots, h_l = g_2^{y_l})$ where $(x_1, \ldots, x_l, y_1, \ldots y_l)$ are chosen randomly. In the challenge query, the challenger computes the public randomness as $u_1 = A^{x_1}, u_2 = A^{x_2}, \ldots, u_l = A^{x_l}$, $v_1 = z^{y_1}, v_2 = z^{y_2}, \ldots, v_l = z^{y_l}$. The challenger can answer leakage queries because it chooses secret key itself.

If the adversary can tell which game he is playing with nonnegligible advantage, then we can conclude that $z = g_2^r$, which breaks the DDH assumption.

Lemma 2. *The output is distributed negligibly close to uniform.*

In Game1, the output can be seen as the multiplication of two-independent leakage resilient one-source extractors in the 2 split-state model. For $l = 2 + (\lambda + 2(\log(1/\varepsilon))/\log N - 3)$ where ε is negligible, the output is the multiplication of two variables which are both distributed ε close to uniform. Thus, it is at least distributed ε close to uniform itself.

Combing lemma 1 and lemma 2, the construction above is a two-source extractor against λ bits leakage under DDH assumption in the 2 split-state model.

4. AFLR-LTFs in 2 Split-State Model

In this section, we formulate the notion of AFLR-LTFs in the 2 split-state model and give concrete constructions of its own and ABO variants.

4.1. Definition. In this model, the secret is divided into 2 parts for storage and leakage attack can only get access to each part independently but not a function of whole state as before. This restriction provides the possibility to achieve AFL resilience.

Definition 5. A collection of 2 split-state ABO-LTFs are composed of specified algorithms as follows:

G_{abo}: the generated trapdoor td is divided into two parts $(\mathrm{td}_1, \mathrm{td}_2)$, as well as the index $s = (s_1, s_2)$. The lossy branch set B_λ^* is the same as before.

F_{abo}^{-1}: the inversion algorithm consists of two subroutines inv_1 and inv_2 which take two parts of the secret as input, respectively. And a combing subroutine f_{-1} takes as input the output of the two subroutines and outputs the preimage.

The security notion requires that index indistinguishability and lossy branch hidden hold even subject to leakage attack. Note that this requirement is just the same as AFLR PKE because the adversary could issue leakage queries to check the lossy branch after it sees the index.

4.2. A Homomorphic AFLR PKE Scheme in 2 Split-State Model. Homomorphism is essential to the underlying PKE schemes for LTFs and CCA security [25]. However, the generic construction in [11] does not incorporate this property. But [11] indicated that variants of hash proof system-based schemes are entropy leakage resilient. So we use the extractors mentioned in Section 3. We start from a basic scheme in [21] which is a variant of the BHHO scheme (and thus hash proof system-based scheme) and then construct the scheme we need via this one.

The basic scheme is as follows:

Gen: choose $x_1, x_2, \ldots, x_l \in \mathbb{Z}_{N-1/4}$, $f_1, f_2, \ldots, f_l \in G_r$. Let $f = \prod_{i=1}^{l} f_i^{x_i}$. The public key is set to be $\mathrm{pk} = (N, f_1, f_2, \ldots, f_l, f)$, and the secret key is $\mathrm{sk} = (x_1, x_2, \ldots, x_l)$.

Enc: given the message $m \in \mathbb{Z}_{N-(1/4)}$, Choose $r \in \mathbb{Z}_{N-(1/4)}$ and compute $(c_1 = (u_1 = f_1^r, u_2 = f_2^r, \ldots, u_l = f_l^r), c_2 = f^r (1+N)^m)$.

Dec: given $c = (c_1, c_2)$. Compute $K = \prod_{i=1}^{l} u_i^{x_i}$ and $m = \log(c_2/K)$.

The construction is as follows:

Given two instances of the basic entropy leakage scheme $(\mathrm{Gen}_1, \mathrm{Enc}_1, \mathrm{Dec}_1)$ and $(\mathrm{Gen}_2, \mathrm{Enc}_2, \mathrm{Dec}_2)$, we define an AFLR PKE scheme in the 2 split-state model as follows:

GEN: it includes two subroutines of $(\mathrm{Gen}_1, \mathrm{Gen}_2)$. The outputs are a public key pair $\mathrm{pk} = (\mathrm{pk}_1, \mathrm{pk}_2)$ and a secret key pair $\mathrm{sk} = (\mathrm{sk}_1, \mathrm{sk}_2)$.

ENC: given a message m, it chooses randomness x_1, x_2, r as the input of two subroutines $(\mathrm{Enc}_1, \mathrm{Enc}_2)$ (we use the same randomness r in both encryption algorithm), and the ciphertext is computed as $C = (C_1, C_2, C_3) = ((c_1^1, c_2^1), (c_1^2, c_2^2), \mathrm{Ext2}(x_1, x_2)(1+N)^m)$.

DEC: $m = \log(C_3/\mathrm{Ext2}(\mathrm{Dec}_1(C_1), \mathrm{Dec}_2(C_2)))$.

Note: discrete logarithm in this case can be easily computed.

Homomorphic property: $\mathrm{ENC}^a(m; r) = \mathrm{ENC}(am; ar)$.

Theorem 2. *The construction above is a CPA secure scheme against* $(1 - O(1))|sk|$ *bits prechallenge leakage and* λ *bits postchallenge leakage under DDH assumption in the 2 split-state model.*

Proof. Let $\mathcal{A}$ denote the adversary. We prove the theorem via a sequence of hybrid experiments as follows.

Game0: the challenger and the adversary proceed as the normal game. The challenger chooses the secret to generate public key and respond leakage queries.

Game1: the only difference from Game0 happens in the challenge phase. The challenger chooses two independent randomness r_1, r_2 for $(\mathrm{Enc}_1, \mathrm{Enc}_2)$ instead of the same one for both.

Game2: in this game, the challenger generates half of public key by itself and runs the simulator of one-entropy leakage resilient instance to get the rest. In detail, the challenger execute Gen to generate $x_1, x_2, \ldots, x_l \in \mathbb{Z}_{N-(1/4)}$, $\mathrm{pk}_1 = (f = \prod_{i=1}^{l} f_i^{x_i}, f_1, f_2, \ldots, f_l)$ and runs a simulator of another instance of the basic scheme to receive $\mathrm{pk}_2 = h, h_1, h_2, \ldots, h_l \in G_r$. The public key is $(N, f_1, f_2, \ldots, f_l, h_1, h_2, \ldots, h_l, f, h)$. When the adversary issues a leakage query $(\mathrm{leak}_1, \mathrm{leak}_2)$, the challenger forward leak_2 to the simulator and receives the answer. The answer can be merged with the output of leak_1 which can be calculated by itself. In the challenge phase, the challenger chooses x_2 and sends it to simulator to get ciphertext (c_1^2, c_2^2). Then, it computes the challenge ciphertext by $C = (C_1, C_2, C_3) = ((c_1^1, c_2^1), (c_1^2, c_2^2), \mathrm{Ext2}(x_1, x_2)(1+N)^m)$. When the adversary issues a postchallenge leakage query, the challenger handles like the way in the prechallenge phase.

Game3: the challenger interacts with $\mathcal{A}$ via two entropy leakage resilient simulators. In this game, all the leakage queries are forwarded to simulators. The challenger computes $\mathrm{Ext2}(x_1, x_2)(1+N)^m$ itself but receives the rest part of ciphertext from simulators. □

Lemma 3. *The views of* $\mathcal{A}$ *in Game0 and Game1 are indistinguishable under DDH assumption.*

This lemma is the same as Theorem 1.

Lemma 4. *The views of* $\mathcal{A}$ *in Game1 and Game2 are indistinguishable following Defintion 4.*

Lemma 5. *The views of* $\mathcal{A}$ *in Game2 and Game3 are indistinguishable following Defintion 4.*

The above two lemmas hold assuming the property of simulator.

Lemma 6. *In Game3, the challenge ciphertext has distribution negligible close to uniform distribution against* $(1 - O(1))|sk|$ *bits prechallenge leakage and* λ *bits postchallenge leakage.*

This can be concluded by the property of the two-source extractor.

4.3. AFLR-LTFS. Following [25], AFLR-ABO-LTFS can be constructed as follows given a homomorphic AFLR-CPA secure encryption scheme $(\mathrm{GEN}, \mathrm{ENC}, \mathrm{DEC})$ which we present above.

PP: choose a branch b as the lossy branch and then run GEN and $\mathrm{ENC}(b) = C$. The public key is (pk, C), and the secret key is sk (we do not put b here because the security is not guaranteed with leaked b, and it can actually be obtained by decrypting C).

Evaluation f: for any input x, choose an evaluation branch b', $f(x) = (C/\mathrm{ENC}(b'))^x = \mathrm{ENC}((b - b')x)$. Output $(f(x), b')$.

Inversion f^{-1}: decrypt C to get b and then compute $f^{-1} = \mathrm{DEC}(f(x))/(b - b')$.

Security analysis: our construction achieves pre- and postchallenge leakage resilience more than [25]. Due to

the use of AFLR encryption scheme as the building block, we can handle leakage query before and after challenge, which makes the proof similar to the one in [25]. So we omit the details here.

Indistinguishability: adversary cannot tell the computation is lossy or not with nonnegligible advantage because the branch is encrypted with the AFLR encryption scheme. If the branch set consists only two elements 0 and 1, this construction can lead to a standard AFLR-LTF which will be used to achieve CCA security later. If the branch set contains many branches, the lossy one is also hidden from adversary.

Lossiness: the output has entropy at most $\log N - 2$. So the lossiness is at least $\log N - (\log N - 2) = 2$. These results can be extended if we use N^a as a module for the basic encryption scheme.

5. Constructions of AFLR-CCA Secure PKE

AFLR-CCA security can be obtained in a classic way with a standard AFLR-LTF, an AFLR-ABO-LTF, and an unforgeable one-time signature scheme. But we prefer another approach via chameleon AFLR-ABO-LTFs. Chameleon ABO-LTFs are introduced in [24] which can avoid using one-time signature. In this variant of LTFs, the lossy set is denoted as a line rather than points to incorporate exponential lossy branches. So we give the construction of chameleon AFLR-ABO-LTFs first.

5.1. Chameleon AFLR-ABO-LTFs

PP: choose $d, e, j \in \mathbb{Z}_{N-1/4}$ and then run GEN and $\mathrm{ENC}(d) = D$, $\mathrm{ENC}(e) = E$, and $\mathrm{ENC}(j) = J$. The public key is (pk, D, E, J), and the secret key is $\mathrm{sk}_{\mathrm{CH}}$.

Evaluation f_{CH}: for any input x, choose an evaluation branch (x_d, x_e), $f_{\mathrm{CH}}(x) = (D^{x_d} E^{x_e} J)^x = \mathrm{ENC}((dx_d + ex_e + j)x)$. Output $(f_{\mathrm{CH}}(x), (x_d, x_e))$.

Inversion f_{CH}^{-1}: decrypt D, E, J to get d, e, j and then compute $f_{\mathrm{CH}}^{-1} = \mathrm{DEC}(f_{\mathrm{CH}}(x))/(dx_d + ex_e + j)$.

The lossy branches are all pairs (x_d, x_e) that satisfy the condition $dx_d + ex_e + j = 0$.

5.2. AFLR-CCA Secure PKE Scheme. We can build our AFLR-CCA secure encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ by combining standard AFLR-LTFs and chameleon AFLR-ABO-LTFs as [24].

$\mathcal{G}$: first generate public parameters for LTF and chameleon ABO-LTF with an AFLR-CPA secure PKE scheme $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$. Let H be a universal hash function and h a collision-resistant hash function. The public key is $(\mathrm{pk}, \mathrm{pk}_{\mathrm{CH}}, H, h, A = \mathrm{Enc}(1), D = \mathrm{Enc}(d), E = \mathrm{Enc}(e), J = \mathrm{Enc}(j))$ where d, e, j are randomly chosen and independently encrypted. Thus, the standard LTF $f(x) = A^x$ and chameleon AFLR-ABO-LTF $f_{\mathrm{CH}}(x; (x_d, x_e)) = (D^{x_d} E^{x_e} J)^x$ are well defined. The secret key is $(\mathrm{sk}, \mathrm{sk}_{\mathrm{CH}})$.

$\mathcal{E}$: for a message m, choose a randomness x, evaluation branch b randomly, and compute $C = (c_0 = H(x) \oplus m, c_1 = f(x), c_2 = f_{\mathrm{CH}}(x; b, h(c_0, c_1)), b)$.

$\mathcal{D}$: given a ciphertext $C = (c_0, c_1, c_2, b)$, compute $x = f^{-1}(c_1)$ and check whether $c_2 = f_{\mathrm{CH}}(x; b, h(c_0, c_1))$. If the output is not $\perp$, then output $m = c_0 \oplus H(x)$.

Theorem 3. *Given AFLR-LTFs and chameleon AFLR-ABO-LTFs, the construction above is an AFLR-CCA secure PKE scheme in the 2 split-state model against λ bits after-the-fact leakage.*

Proof sketch: the case without leakage attack are proven secure in [24]. The proof goes with a sequence of indistinguishable games between challenger and adversary. The first step is to reject all the decryption queries with lossy computation by chameleon ABO-LTF. Then, change the working mode of LTF to be lossy and the decryption queries can be responded by chameleon LTF on injective branches. Finally, CCA security is achieved statistically with appropriate parameter.

As the underlying AFLR primitives we propose in Sections 4 and 5.1 can handle leakage queries in both pre- and postchallenge phase, we can preserve AFLR security if we use these primitives instead of ones in ordinary case naturally. Readers can check every step and see the proof strategy above can still work with additional leakage attack. □

6. Efficiency in Practice

The generic constructions in previous works [15, 17] need NIZK system to prove the language that two encryptions contain the same plaintext. In practice, NIZK proofs secure in standard model concerns the Groth–Sahai system [26] which suffers from heavy burden of computations via bilinear mappings. Specifically, proving a commitment of exponential which is only a step stone for proving equal plaintext requires 4 group elements and verified by 9 pairing operations. The cost of NIZK for same plaintext may be dozens of group elements and pairing operations. That is why existing works did not even give concrete construction for NIZK-based solutions. This situation is just like "two-key" generic construction in [16] which is convincing but not practical until [19] appeared. Our construction comes from a leakage resilient extension of [19] and achieves CCA security against postleakage without NIZK just like Cramer and Shoup [19] did in classic environment.

Specifically, the evaluation key in our scheme can be processed in precomputation and the encryption algorithm works by $8l$ exponential computations. If we want to achieve 80 bit security ($\varepsilon = 2^{-80}$) with 1024 bit N, $l = 2 + ((\lambda + 160)/7)$ against λ bit leakage. If we want to encrypt longer plaintext, we can use larger modulus like $N^a, a > 2$.

We implement our scheme to evaluate its efficiency, which is based on JPBC 2.0.0 library (http://gas.dia.unisa.it/projects/jpbc/index.html#.VTDrLSOl_Cw) and coding

language Java. We select type A1 pairings are constructed on the curve $y^2 = x^3 + x$ over the field $\mathbb{Z}_N$ for some Blum integer N. The following experiments are based on Dell laptop (Windows 7 operation system with Intel(R) Core(TM) i5-2450M CPU 2.50 GHz, 4.00 GB RAM, and 500 G disk storage). The time cost in real-world experiment for one encryption is 0.042 s with 1024 bit N.

7. Conclusion and Future Direction

Our work removes the use of zero knowledge proofs which is not efficient in the construction of AFLR-CCA secure PKE encryption schemes via the approach of lossy trapdoor functions. We also present instances of AFLR-LTF and its variants. An interesting open problem is finding more efficient PKE schemes with both homomorphic property and leakage resilience.

Data Availability

The simulation data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key R&D Program of China (2017YFB0802000), the National Natural Science Foundation of China (61572303 and 61772326), the Natural Science Basic Research Plan in Shaanxi Province of China (2018JQ6088), the National Cryptography Development Fund during the 13th Five-year Plan Period (MMJJ20170216), the Foundation of State Key Laboratory of Information Security (2017-MS-03), the Fundamental Research Funds for the Central Universities (GK201702004 and GK201803064), and the Project of Basic Research of Qinghai Province (2016-ZJ-776).

References

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan, "Simultaneous hardcore bits and cryptography against memory attacks," *Theory of Cryptography*, pp. 474–495, 2009.

[2] M. Noar and G. Segev, "Public-key cryptosystems resilient to key leakage," *SIAM Journal on Computing*, vol. 41, no. 4, pp. 772–814, 2012.

[3] E. Boyle, G. Segev, and D. Wichs, "Fully leakage-resilient signatures," *Journal of Cryptology*, vol. 26, no. 3, pp. 513–558, 2013.

[4] A. Faonio, J. B. Nielsen, and D. Venturi, "Fully leakage-resilient signatures revisited: graceful degradation, noisy leakage, and construction in the bounded-retrieval model," *Theoretical Computer Science*, vol. 660, pp. 23–56, 2017.

[5] J.-D. Wu, Y.-M. Tseng, and S.-S. Huang, "Leakage-resilient ID-based signature scheme in the generic bilinear group model," *Security and Communication Networks*, vol. 9, no. 17, pp. 3987–4001, 2016.

[6] J. Alwen, Y. Dodis, and D. Wichs, "Leakage-resilient public-key cryptography in the bounded-retrieval model," *Advances in Cryptology—CRYPTO 2009*, vol. 5677, pp. 36–54, 2009.

[7] J. Alawatugoda, "On the leakage-resilient key exchange," *Journal of Mathematical Cryptology*, vol. 11, no. 4, pp. 215–269, 2017.

[8] B. Qin, K. Chen, and S. Liu, "Efficient chosen-ciphertext secure public-key encryption scheme with high leakage-resilience," *IET Information Security*, vol. 9, no. 1, pp. 32–42, 2015.

[9] Y. Zhou, B. Yang, W. Zhang, and Y. Mu, "CCA2 secure public-key encryption scheme tolerating continual leakage attacks," *Security and Communication Networks*, vol. 9, no. 17, pp. 4505–4519, 2016.

[10] S.-F. Sun, D. Gu, and S. Liu, "Efficient chosen ciphertext secure identity-based encryption against key leakage attacks," *Security and Communication Networks*, vol. 9, no. 11, pp. 1417–1434, 2016.

[11] S. Halevi and H. Lin, "After-the-fact leakage in public-key encryption," in *Theory of Cryptography*, pp. 107–124, Springer, Berlin, Germany, 2011.

[12] S. Micali and L. Reyzin, "Physically observable cryptography," in *Theory of Cryptography*, pp. 278–296, Springer, Berlin, Germany, 2004.

[13] J. Li, Y. Guo, Q. Yu, Y. Lu, and Y. Zhang, "Provably secure identity-based encryption resilient to post-challenge continuous auxiliary input leakage," *Security and Communication Networks*, vol. 9, no. 10, pp. 1016–1024, 2016.

[14] Z. Yang and S. Li, "On security analysis of an after-the-fact leakage resilient key exchange protocol," *Information Processing Letters*, vol. 116, no. 1, pp. 33–40, 2016.

[15] Z. Zhang, S. S. M. Chow, and Z. Cao, "Post-challenge leakage in public-key encryption," *Theoretical Computer Science*, vol. 572, pp. 25–49, 2015.

[16] M. Noar and M. Yung, "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing—STOC'90*, pp. 427–437, Baltimore, MD, USA, May 1990.

[17] S. Chakraborty, G. Paul, and C. P. Rangan, "Efficient compilers for after-the-fact leakage: from CPA to CCA-2 secure PKE to AKE," in *Information Security and Privacy*, pp. 343–362, Springer, Berlin, Germany, 2017.

[18] E. Fujisaki, A. Kawachi, R. Nishimaki, K. Tanaka, and K. Yasunaga, "Post-challenge leakage resilient public-key cryptosystem in split state model," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E98.A, no. 3, pp. 853–862, 2015.

[19] R. Cramer and V. Shoup, "Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption," in *Advances in Cryptology—EUROCRYPT 2002*, pp. 45–64, Springer, Berlin, Germany, 2002.

[20] C. Peikert and B. Waters, "Lossy trapdoor functions and their applications," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1803–1844, 2011.

[21] B. Qin, S. Liu, K. Chen, and M. Charlemagne, "Leakage-resilient lossy trapdoor functions and public-key encryption," in *Proceedings of the First ACM Workshop on Asia Public-Key Cryptography—AsiaPKC'13*, pp. 3–12, Hangzhou, China, May 2013.

[22] Y. Chen, B. Qin, and H. Xue, "Regular lossy functions and their applications in leakage-resilient cryptography," *Theoretical Computer Science*, vol. 739, pp. 13–38, 2018.

[23] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky, "Circular-secure encryption from decision diffie-hellman," in *Proceedings of the CRYPTO*, pp. 108–125, Santa Barbara, CA, USA, August 2008.

[24] S. Liu, J. Lai, and R. H. Deng, "General construction of chameleon all-but-one trapdoor functions," *Journal of Internet Services and Information Security*, vol. 1, no. 2-3, pp. 74–88, 2011.

[25] B. Hemenway and R. Ostrovsky, "Homomorphic encryption over cyclic groups implies chosen-ciphertext security," *IACR Cryptology*, vol. 99, 2010.

[26] J. Groth, "Simulation-sound NIZK proofs for a practical language and constant size group signatures," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 444–459, Springer, Shanghai, China, December 2006.