Public-Key Encryption with Fuzzy Keyword Search withstanding Keyword Guessing Attack

Peng Xu and Hai Jin, Senior Member, IEEE

***Abstract*—With tremendous interests having been attracted, public-key encryption with keyword search (PEKS) achieves the searchability of encrypted keywords without sharing a secret between senders and receivers. However, nearly all existing PEKS schemes and the expansions obtain their provable security under an implicit condition that the size of the targeted keyword space is beyond the polynomial level, which shows inefficiency in practice. Hence, these existing schemes are insecure under keyword guessing attack in applications. As we observe, the key to defend the attack is to avoid the availability of the exact keyword trapdoor to adversaries. Accordingly, we compromise the exactness of keyword trapdoor by mapping at least two different keywords into a common fuzzy keyword trapdoor.**

In this paper, we first propose a novel concept called public-key encryption with fuzzy keyword search (PEFKS), in which searchers only obtain the fuzzy keyword trapdoor instead of the exact keyword trapdoor in PEKS. Subsequently, for the keyword space with and without uniform entropy, we respectively present two universal transformations from anonymous identity-based encryption (IBE) to PEFKS as well as an instance in the uniform case. Moreover, their provable security are presented respectively under adaptive-ID and chosen plaintext attacks and keyword guessing attack. As the best we know, PEFKS is the first and promising scheme against keyword guessing attack under the practical condition that the keyword space is not more than the polynomial level.

***Index Terms*—Public-key encryption with keyword search, keyword guessing attack, public-key encryption with fuzzy keyword search, anonymous identity-based encryption**

I. Introduction

PUBLIC-KEY encryption with keyword search (PEKS) [1] is a novel cryptographic approach to keep the privacy of keyword by public-key encryption, as well as to guarantee the efficient keyword search in ciphertexts. Even before PEKS was first proposed by Boneh et al. in 2004 [1], tremendous efforts had been devoted to explore efficiently searching encrypted datum already, since the significance is conspicuous to support the privacy and searchability of datum in incredible databases. Hence, several appreciable works were made in the recent decade. In 2000, Song et al. proposed a practical and provably secure techniques for searching encrypted data [2]. With regard to the performance of search, Agrawal et al. proposed an order preserving encryption [3]. Based on the work of Song et al.'s, Curtmola et al. proposed an advanced scheme for an enhanced performance [4]. Referring to these works, we found that the encryption and searchability of datum should rely on sharing secrets between sends and receivers. Consequently, these works, except PEKS, will be seriously doubted in practice with difficulties on establishing and managing these shared secrets. In contrast, PEKS is exceedingly convenient in practice without sharing secrets.

As Boneh et al. first proposed PEKS, he also proposed a universal transformation from anonymous identity-based encryption (IBE) [1] [5, 6, 7, 8] to PEKS [1]. Hereafter, PEKS has been given tremendous attention. Furthermore, Abdalla et al., who completed the foundations of PEKS, presented an improved universal transformation from anonymous IBE to PEKS and a novel expansion of PEKS that public-key encryption with temporary keyword search (PETKS) in 2005. To achieve combinable multi-keyword search, two schemes on public-key encryption with conjunctive keyword search (PECKS) [9, 10] were respectively proposed in 2004 and 2007. Conclusively, the aforementioned schemes have a common character that they only succeeded on the equality search, rather than achieved range search and so on. Hence, Bethencourt et al. succeeded on public-key encryption with conjunctive keyword range search [11] by anonymous hierarchical IBE (HIBE) [6] in 2006, and further updated their work in 2007 [12]. In TCC'2007 , Boneh et al. proposed a novel technique called hidden vector encryption (HVE) to achieve conjunctive, range and subset searches [13]. In addition, an improved trapdoor generation of keywords was proposed by Camenisch et al. [14], who employed the committed two-part computation protocol and achieved the invisibility of keyword to generator. The corresponding scheme was called public-key encryption with oblivious keyword search (PEOKS). Although several efficiently conjunctive keyword search over encrypted datum were proposed [15, 16], they need sharing a secret between sends and receivers. Hence, they were troublesome in application. Conclusively, all researches on PEKS focused on the various searchability in recent years. However, its practical security is almost out of consideration.

So far, all of proposed PEKS schemes and the expansions proved their security in some sense. However, their provable security were commonly based on an implicit condition that the size of keyword space must be beyond the polynomial level. Therefore any adversary bounded by the computational ability can not efficiently and exhaustively search the keyword space to successfully guess keywords in PEKS. Nevertheless,

P. Xu and H. Jin are with Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China. E-Mail: xupeng@mail.hust.edu.cn, hjin@hust.edu.cn.

[1]The first anonymous IBE scheme was proposed by Boneh et al. in 2001 and proved the security in the random oracle (RO) model [5]. In 2006, Boyen et al. proposed an anonymous IBE scheme [6], which first has provable security in the standard model. In the same year, Gentry proposed the most efficient anonymous IBE scheme so far [7]. Ducas proposed an anonymous IBE scheme first based on the asymmetric bilinear map in 2010 [8].

we confirm that the implicit condition is obviously unreasonable in practice from two sides:

- In practice, keywords would be semantical and indexed in a dictionary. Moreover, somebody would like to use the common keywords, such as 'Urgency'. Hence, keywords may be non-uniformly employed. In addition, the keyword space should be carefully determined at the initiation of cryptosystem [17]. Otherwise, for the keywords having the same meaning, such as 'urgent' and 'imperative', different people may employed different keywords. It induces that searchers need several search trapdoors for the same meaning, and the time cost of search will be multiplied. Hence, when we need to initially determine the keyword space, the necessity is that the size of it is not more than the polynomial level.
- In theory, it is unreasonable to assume that the size of keyword space is the exponential level. Giving a counter example that assuming the size of keyword space is 2^k, and $\mathcal{E}_{K^i}$ bits is the entropy of keyword K^i, we trivially have

$$\sum_{i=1}^{2^k} 2^{-\mathcal{E}_{K^i}} = 1 \tag{1}$$

Let $Poly()$ denote any polynomial. If all keywords have $2^{-\mathcal{E}_{K^i}} \geq Poly(k)$, it obviously has $\sum_{i=1}^{2^k} 2^{-\mathcal{E}_{K^i}} \gg 1$, which is contrary with Equation 1. Hence, in the keyword space, there exists keywords satisfying $2^{-\mathcal{E}_{K^i}} < Poly(k)$. Moreover, the number of keywords satisfying $2^{-\mathcal{E}_{K^i}} > Poly(k)$ is not more than $Poly(k)$. In other words, the number of keywords having the practical probability to be employed is not more than $Poly(k)$, even if the keyword space is the exponential level.

Under the practical condition that the size of keyword space is not more than the polynomial level, Byun et al. first proposed keyword guessing attack [11] and attacks a PEKS scheme and a PECKS scheme [9] successfully in 2006. Under the same condition Jeong et al. proved that any PEKS scheme satisfying at least computationally indistinguishable consistency implies successful keyword guessing attack [18] necessarily. Moreover, since satisfying at least computationally indistinguishable consistency is necessary for any efficient PEKS, it seems impossible to defend keyword guessing attack under the practical condition.

A. the Motivation

Referring to PEKS, we found that to complete a keyword guessing attack, an adversary needs a trapdoor of the keyword first, then successfully guess this keyword by a brute force way. Hence, the key of defending keyword guessing attack is to avoid that adversaries know the trapdoor of the exact keyword.

We heuristically conceived a passive scheme that public-key encryption without keyword search (called Passive-PEKS). Obviously, no search trapdoor of keyword is needed in the passive scheme. Moreover, keywords are encrypted by using the existing cryptosystem to keep the privacy. Hence, no adversary can successfully attack the scheme. However, assuming we want to provide the keyword searchability in Passive-PEKS, we need to provide the search trapdoor of keywords. Moreover, when providing the trapdoor of the exact keyword, Passive-PEKS appropriately functions as PEKS such that the content of keywords is leaked under keyword guessing attack. Conclusively, we either achieve the keyword searchability, or maintain the security of keywords. Hence, we were motivated to propose a method or scheme to tradeoff the searchability and security, such that the searchability can be achieved as well as possible without losing the security of keywords.

Consequently, we were motivated to propose a fuzzy keyword search, which is the first solution to defend keyword guessing attack in PEKS as the best we know. In the fuzzy keyword search, adversaries only know the fuzzy trapdoor of keywords. Hence, they can not deterministically guess keywords. Moreover, taking advantage of the entropy of the keyword space, we will further reduce the biased advantage of successful guess in the best effort.

B. Our Contributions

According to the motivation, public-Key encryption with fuzzy keyword search (PEFKS) will be novelly define in our paper. Furthermore, we will make two universal transformations from anonymous IBE to PEFKS respectively under different conditions and prove their security under adaptive-ID and chosen plaintext attacks and keyword guessing attack. Specifically, we will firstly present a universal and provably secure transformation from anonymous IBE to PEFKS for the keyword space of uniform entropy (called PEFKS-UE). Moreover, based on the anonymous IBE scheme proposed by Boneh in 2001 [5], we will propose a provably secure instance of PEFKS-UE. Secondly, we will present a universal and provably secure transformation from anonymous IBE to PEFKS for the keyword space of non-uniform entropy (called PEFKS-NE). In addition, we will illuminate that the biased advantage in keyword guessing attack has been reduced as much as possible in PEFKS-NE.

C. Organization

The organization of this paper is as follows. In Section II, some preliminary definitions and our definition of PEFKS will be given. In Section III, it presents the universal transformation PEFKS-UK and an instance of it, then prove their security. Section IV presents the universal transformation PEFKS-NE, its provable security and an important analysis. Section V presents conclusion.

II. Preliminaries

In this section, several definitions will be presented for simplifying descriptions of our contributions. Throughout this paper we employ $Poly()$ to denote any polynomial.(Note that unless stated otherwise, all symbols have the same meaning as when they first appear in this paper.)

A. Anonymous IBE

We redefine the previous work on IBE [5] and anonymous IBE [6] as follows.

Definition 1 (IBE)**.** *IBE consists of the following polynomial time algorithms:*

- $Setup(k, r_1)$*: Take as input a security parameter* k *and a random tape* r_1*, then generate a pair of public-and-private system parameters* Pub_{IBE} *and* Pri_{IBE}*. Moreover,* Pub_{IBE} *includes the message space* $\mathcal{M}$*, the ciphertext space* $\mathcal{C}$ *and the identity space* $\mathcal{ID}$*.*
- $Extract(Pri_{IBE}, r_2, ID)$*: Take as input a private system parameter* Pri_{IBE}*, an identity* $ID \in \mathcal{ID}$ *and a random tape* r_2*, then generate the private key* $PriK_{ID}$ *of* ID*.*
- $Encrypt(Pub_{IBE}, r_3, ID, M)$*: Take as input a public system parameters* Pub_{IBE}*, a receiver's identity* $ID \in \mathcal{ID}$*, a random tape* r_3 *and a message* $M \in \mathcal{M}$*, then generate a ciphertext* $C \in \mathcal{C}$*.*
- $Decrypt(Pub_{IBE}, PriK_{ID}, C)$*: Take as input a public system parameters* Pub_{IBE}*, the receiver's private key* $PriK_{ID}$ *and a ciphertext* $C \in \mathcal{C}$*, then return the decryption result of* C*.*

Moreover, it satisfies the consistency that for any ciphertext $C = Encrypt(Pub_{IBE}, r_3, ID', M)$*,* $M = Decrypt(Pub_{IBE}, PriK_{ID}, C)$ *holds if and only if* $ID = ID'$*, where* ID *is randomly chosen in* $\mathcal{ID}$*.*

Definition 2 (Anonymous IBE)**.** *An anonymous IBE is a kind of IBE that given any valid ciphertext, no probabilistically polynomial time adversary has non-negligible advantage to decide which identity was used to generate the ciphertext.*

B. PEKS and its Keyword Guessing Attack

We also redefine PEKS [1] and its keyword guessing attack [19] as follows.

Definition 3 (PEKS)**.** *PEKS consists of the following polynomial time algorithms:*

- $SysG(k, r_1)$*: Take as input a security parameter* k *and a random tape* r_1*, then generate a pair of public-and-private system parameters* Pub_{PEKS} *and* Pri_{PEKS}*. Moreover,* Pub_{PEKS} *includes the keyword space* $\mathcal{K}$*.*
- $Trapdoor(Pri_{PEKS}, r_2, K)$*: Take as input a private system parameter* Pri_{PEKS}*, a random tape* r_2 *and a keyword* $K \in \mathcal{K}$*, then generate a search trapdoor* T_K*.*
- $CipherG(Pub_{PEKS}, r_3, K)$*: Take as input a public system parameters* Pub_{PEKS}*, a random tape* r_3 *and a keyword* $K \in \mathcal{K}$*, then generate a searchable ciphertext of* K*.*
- $ExactTest(Pub_{PEKS}, T_K, C)$*: Take as input a public system parameters* Pub_{PEKS}*, a search trapdoor* T_K *and a searchable ciphertext* $C = CipherG(Pub_{PEKS}, K')$*, then return* B*, where*

$$B = \begin{cases} 1 & if\ K' = K; \\ 0 & otherwise. \end{cases} \tag{2}$$

Moreover, it satisfies the consistency that for any keyword searchable ciphertext $C' = CipherG(Pub_{PEKS}, K')$*,* $ExactTest(Pub_{PEKS}, T_K, C')$ *returns '1' if and only if* $K = K'$*, where* K *is randomly chosen in* $\mathcal{K}$*.*

Definition 4 (Keyword Guessing Attack on PEKS)**.** *Given a public system parameters* Pub_{PEKS} *and a valid trapdoor* T_K*, the adversary indexes all keywords in the keyword space* $\mathcal{K}$ *as* $\{K^1, K^2, \ldots, K^{|\mathcal{K}|}\}$ *and implements keyword guessing attack as follows:*

1) *Let* $i = 1$*.*
2) *Generate a keyword searchable ciphertext* C *of* K^i*, where* $C = CipherG(Pub_{PEKS}, r_3, K^i)$*.*
3) *If* $ExactTest(Pub_{PEKS}, T_K, C) = 1$*, return* K^i*;*
4) *If* $i \neq |\mathcal{K}|$*, compute* $i = i+1$ *and go to step 2; otherwise it returns '*$\perp$*' (it means the abortion).*

According to the consistency of PEKS [2], the adversary's output K^i equals to K with at most negligible error probability. Specifically, once adversary had a valid trapdoor, according to the consistency of PEKS, he can exhaustively search all keywords and deterministically guess the keyword that is used to generate the trapdoor. Moreover, this process is efficient under the practical condition that $|\mathcal{K}| \leq Ploy(k)$.

C. PEFKS and its Keyword Guessing Attack

Definition 5 (PEFKS)**.** *PEFKS consists of the following polynomial time algorithms:*

- $SysG(k, r_1)$*: Take as input a security parameter* k *and a random tape* r_1*, then generate a pair of public-and-private system parameters* Pub_{PEFKS} *and* Pri_{PEFKS}*. Moreover,* Pub_{PEFKS} *includes the keyword space* $\mathcal{K}$ *and a deterministic function* $Fuz(K, \mathcal{K})$*, which takes as input a keyword* K *and the keyword space* $\mathcal{K}$*, then deterministically returns a fuzzy value such that there are at least two different keywords have the same fuzzy value.*
- $DTrapdoor(Pri_{PEFKS}, r_2, r_2', K)$*: Take as input a private system parameter* Pri_{PEFKS}*, two random tapes* r_2 *and* r_2'*, and a keyword* $K \in \mathcal{K}$*, then compute fuzzy value* $FK = Fuz(K, \mathcal{K})$*, finally return a fuzzy search trapdoor* FT_K *for* FK *and an exact search trapdoor* ET_K *for* K*. Note that, on one hand keywords with the same fuzzy value necessarily have the same fuzzy search trapdoor; on the other hand, all distinct keywords have the same exact search trapdoor with at most the negligible probability.*
- $CipherG(Pub_{PEFKS}, r_3, K)$*: Take as input a public system parameters* Pub_{PEFKS}*, a random tape* r_3 *and a keyword* $K \in \mathcal{K}$*, then generate a fuzzy keyword searchable ciphertext of* K*.*
- $FuzzTest(Pub_{PEFKS}, FT_K, C)$*: Take as input a public system parameters* Pub_{PEFKS}*, a fuzzy search trapdoor* FT_K *and a fuzzy keyword searchable ciphertext* $C = CipherG(Pub_{PEFKS}, r_3, K')$*, then return* B*, where*

$$B = \begin{cases} 1 & if\ Fuz(K', \mathcal{K}) = Fuz(K, \mathcal{K}); \\ 0 & otherwise. \end{cases} \tag{3}$$

[2]The consistency is computationally indistinguishable at least[20].

- $ExactTest(Pub_{PEFKS}, ET_K, C)$*: Take as input a public system parameters* Pub_{PEFKS}*, an exact search trapdoor* ET_K *and a fuzzy keyword searchable ciphertext* $C = CipherG(Pub_{PEFKS}, r_3, K')$*, then return* B*, where*

$$B = \begin{cases} 1 & if\ K' = K; \\ 0 & otherwise. \end{cases} \tag{4}$$

Moreover, it satisfies the consistency that for any fuzzy keyword searchable ciphertext $C' = CipherG(Pub_{PEFKS}, r_3, K')$*, algorithm* $FuzzTest(Pub_{PEFKS}, FT_K, C')$ *returns '1' if and only if* $Fuz(K', \mathcal{K}) = Fuz(K, \mathcal{K})$*; algorithm* $ExactTest(Pub_{PEFKS}, ET_K, C')$ *returns '1' if and only if* $K = K'$*, where* K *and* K' *are randomly chosen in* $\mathcal{K}$*.*

Referring to the definition of PEFKS, the principal difference between PEFKS and PEKS is the fuzzy test/search algorithm $FuzzTest$, which was not included in PEKS. In practice, the server, who stores the encrypted datum, implements the exact test/search in PEKS. However, in PEFKS the server only receives a fuzzy search trapdoor of the keyword from a receiver at first; secondly, it implements the fuzzy test/search; at last, it sends back all of satisfying ciphertexts to the receiver. Specially, the exact test/search algorithm $ExactTest$ only be implemented by the receiver to decide which ciphertext contains the exact keyword. Based on the fact that nobody except the receiver knows the exact search trapdoors in PEFKS, we adaptively define the keyword guessing attack on PEFKS as follows.

Definition 6 (Keyword Guessing Attack on PEFKS)**.** *Given a public system parameters* Pub_{PEFKS} *and a fuzzy search trapdoor* FT_K*, the adversary indexes all keywords in the keyword space* $\mathcal{K}$ *as* $\{K^1, K^2, \ldots, K^{|\mathcal{K}|}\}$ *and implements keyword guessing attack as follows:*

1) *Let* $i = 1$.
2) *Generate a keyword searchable ciphertext* C *of* K^i*, where* $C = CipherG(Pub_{PEKS}, r_3, K^i)$.
3) *If* $FuzzTest(Pub_{PEFKS}, FT_K, C) = 1$*, output* K^i.
4) *If* $i \neq |\mathcal{K}|$*, compute* $i = i+1$ *and go to step 2; otherwise it returns '*$\perp$*'.*

Referring to Definition 6, since at least two different keywords have the same fuzzy value, to implement keyword guessing attack on PEFKS, the adversary gets at least two different keywords. Hence, he fails to deterministically decide which keyword is the exact one that was used to generate the fuzzy search trapdoor. Furthermore, since nobody except the receiver knows the exact trapdoors of keywords, the adversary also fails to implement the keyword guessing attack, which is analogous with Definition 4. Conclusively, we intuitively decide that PEFKS can defend keyword guessing attack.

III. PEFKS-UE

Let k be a security parameter. Let $\sum$ be an alphabet. Let the keyword space $\mathcal{K} = \sum^n$, where $|\mathcal{K}| \leq Ploy(k)$. Moreover, $\mathcal{K}$ has the uniform entropy such that each symbol of $\sum$ has the identical or computationally indistinguishable probability to be used. In PEFKS-UE, the function $Fuz(K, \mathcal{K})$ is redefined as follows:

1) Take as input a keyword $K \in \mathcal{K}$, then parse K as $K = K_1 || \ldots || K_n$ where $K_i \in \sum$ for $i \in [1, n]$.
2) Finally return $FK = K_1 || \ldots || K_{(n-1)}$.

The complete description of PEFKS-UE is as follows.

A. A Universal Transformation

Let $UIBE = (Setup, Extract, Encrypt, Decrypt)$ be a universal anonymous IBE defined in Definition 1. Let $H_1 : \mathcal{FK} \bigwedge \mathcal{K} \rightarrow \mathcal{ID}$ be a collision resistant function. PEFKS-UE consists of the following algorithms:

- $SysG(k, r_1)$: Take as input a security parameter k and a random tape r_1, then run algorithm $Setup(k, r_1)$ of $UIBE$ to generate a pair of public-and-private system parameters that

$$\begin{aligned} Pub_{PEFKS-UE} &= \langle Pub_{UIBE}, Fuz, H_1, \mathcal{K} \rangle \\ Pri_{PEFKS-UE} &= Pri_{UIBE} \end{aligned} \tag{5}$$

- $DTrapdoor(Pri_{PEFKS-UE}, r_2, r_2', K)$: Take as input a private system parameter $Pri_{PEFKS-UE} = Pri_{UIBE}$, two random tapes r_2 and r_2', and a keyword $K \in \mathcal{K}$, then generate a fuzzy search trapdoor FT_K and an exact search trapdoor ET_K, where

$$\begin{aligned} FT_K &= Extract(Pri_{UIBE}, r_2', H_1(Fuz(K, \mathcal{K}))) \\ ET_K &= Extract(Pri_{UIBE}, r_2, H_1(K)) \end{aligned} \tag{6}$$

- $CipherG(Pub_{PEFKS-UE}, r_3, r_3', K)$: Take as input a public system parameters $Pub_{PEFKS-UE}$, two random tapes r_3 and r_3', and a keyword $K \in \mathcal{K}$, then randomly choose a message $M \in \mathcal{M}$, finally generate a fuzzy keyword searchable ciphertext $\langle M, C_F, C_E \rangle$, where

$$\begin{aligned} C_F &= Encrypt(Pub_{UIBE}, r_3, H_1(Fuz(K, \mathcal{K})), M) \\ C_E &= Encrypt(Pub_{UIBE}, r_3', H_1(K), M) \end{aligned} \tag{7}$$

- $FuzzTest(Pub_{PEFKS-UE}, FT_K, \langle M, C_F \rangle)$: Take as input a public system parameters $Pub_{PEFKS-UE}$, a fuzzy search trapdoor FT_K and a searchable ciphertext $\langle M, C_F \rangle$, then return B, where

$$B = \begin{cases} 1 & if\ M = Decrypt(Pub_{UIBE}, FT_K, C_F); \\ 0 & otherwise. \end{cases} \tag{8}$$

- $ExactTest(Pub_{PEFKS-UE}, ET_K, \langle M, C_E \rangle)$: Take as input a public system parameters $Pub_{PEFKS-UE}$, an exact search trapdoor ET_K and a searchable ciphertext $\langle M, C_E \rangle$, then return B, where

$$B = \begin{cases} 1 & if\ M = Decrypt(Pub_{UIBE}, ET_K, C_E); \\ 0 & otherwise. \end{cases} \tag{9}$$

The consistency of PEFKS-UE. According to Theorem 4.2 in [20], we easily find that the aforementioned PEFKS-UE is consistent, when the anonymous IBE scheme $UIBE$ satisfies the semantic security.

B. An Instance of PEFKS-UE

Based on the anonymous IBE scheme proposed by Boneh ea al. [5] (called BF01 scheme in this paper), we construct an instance of PEFKS-UE according to the universal transformation PEFKS-UE as follows:

Let $\mathbb{G}$ and $\mathbb{G}_t$ denote two multiplicative groups of prime order q. Let g be a generator of $\mathbb{G}$. Let the bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_t$ [5, 21, 22] be an efficiently computable and non-degenerate function with the bilinearity that $e(g^a, g^b) = e(g, g)^{ab}$, where a and b are randomly chosen in $\mathbb{Z}_q^*$ and $e(g, g)$ is the generator of $\mathbb{G}_t$. Let the bilinear map generator $BGen(1^k)$ be an efficient algorithm that given a security parameter k, it returns $\langle q, \mathbb{G}, \mathbb{G}_t, g, e\rangle$. In addition, let the symbol $\stackrel{R}{\leftarrow}$ denote randomly choosing an element (or elements) at the left side from the field at the right side.

- $SysG(k, r_1)$: Given a security parameter k and a random tape r_1, this algorithm works as follows:
  1) Run $BGen(1^k)$ to generate $\langle q, \mathbb{G}, \mathbb{G}_t, g, e\rangle$.
  2) Set $g_{pub} = g^s$, where $s \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$.
  3) Choose a collision resistance function $H_1 : \sum^{n-1} \bigvee \mathcal{K} \rightarrow \mathbb{G}$.
  4) Choose a cryptographic hash function $H_2 : \mathbb{G}_t \rightarrow \mathcal{M}$.

  The keyword space is $\mathcal{K} = \sum^n$. The message space is $\mathcal{M} = \{0, 1\}^{k_1}$. The public system parameters and the private system parameter are as follows:

$$\begin{aligned} Pub_{PEFKS-UE} =& \langle q, \mathbb{G}, \mathbb{G}_t, g, e, g_{pub}, Fuz \\ & H_1, H_2, \mathcal{K}, \mathcal{M}\rangle \\ Pri_{PEFKS-UE} =& s \end{aligned} \tag{10}$$

- $DTrapdoor(Pri_{PEFKS-UE}, K)$: Given a private system parameter $Pri_{PEFKS-UE}$ and a keyword K, this algorithm works as follows:
  1) Compute $g_{FK} = H_1(Fuz(K, \mathcal{K})) \in \mathbb{G}$, then output the fuzzy search trapdoor $FT_K = g_{FK}^s$.
  2) Compute $g_{EK} = H_1(K) \in \mathbb{G}$, then return the exact search trapdoor $ET_K = g_{EK}^s$.
- $CipherG(Pub_{PEFKS-UE}, r_3, r_3', K)$: Given a public system parameters $Pub_{PEFKS-UE}$, two random tapes r_3 and r_3' and a keyword K, this algorithm works as follows:
  1) Randomly choose a message $M \stackrel{R}{\leftarrow} \mathcal{M}$ and two numbers $\langle t, t'\rangle \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$.
  2) Compute $g_{FK} = H_1(Fuz(K, \mathcal{K}))$ and $g_{EK} = H_1(K)$.
  3) Return a fuzzy keyword searchable ciphertext $\langle M, C_F, C_E\rangle$, where

$$\begin{aligned} C_F &= \langle g^{t'}, H_2(e(g_{FK}, g_{pub})^{t'}) \oplus M\rangle \\ C_E &= \langle g^t, H_2(e(g_{EK}, g_{pub})^t) \oplus M\rangle \end{aligned} \tag{11}$$

- $FuzzTest(Pub_{PEFKS-UE}, FT_K, \langle M, C_F\rangle)$: Given a public system parameters $Pub_{PEFKS-UE}$, a fuzzy search trapdoor FT_K and a part of a fuzzy keyword searchable ciphertext that $\langle M, C_F\rangle$, then parse C_F as $\langle C_{F1}, C_{F2}\rangle$, finally return B, where

$$B = \begin{cases} 1 & if\ M = C_{F2} \oplus H_2(e(FT_K, C_{F1})); \\ 0 & otherwise. \end{cases} \tag{12}$$

- $ExactTest(Pub_{PEFKS-UE}, ET_K, \langle M, C_E\rangle)$: Given a public system parameters $Pub_{PEFKS-UE}$, an exact search trapdoor ET_K and a part of the searchable ciphertext that $\langle M, C_E\rangle$, then parse C_E as $\langle C_{E1}, C_{E2}\rangle$, finally return B, where

$$B = \begin{cases} 1 & if\ M = C_{E2} \oplus H_2(e(ET_K, C_{E1})); \\ 0 & otherwise. \end{cases} \tag{13}$$

On one hand, according to the IND-ID-CPA (indistinguishability of ciphertexts under adaptive-ID and chosen plaintext attacks) security of the BF01 scheme, it is easily to implies that the instance of PEFKS-UE has the computationally indistinguishable consistency. On the other hand, in the instance we avoided constructing it rigidly according to the universal transformation PEFKS-UE, whereas some tricks were employed to improve the performance. For example, the fuzzy keyword searchable ciphertext has the shorter size.

C. Security proofs

Under the practical condition that $|\mathcal{K}| \leq Poly(k)$, we respectively prove the security of the instance of PEFKS-UE and the universal transformation PEFKS-UE. Specially, their security under keyword guessing attack is also considered. The proofs are as follows:

- *The security of the instance of PEFKS-UE.* On one hand, we proved that the instance of PEFKS-UE inherits the Anon-ID-CPA [23, 6] (anonymity of ciphertext under adaptive-ID and chosen plaintext attacks) security of the BF01 scheme IBE by the reduction proof in Table I. Hence, for any keyword $K \in \mathcal{K}$, anybody without the exact search trapdoor of K can not know its content from the fuzzy keyword searchable ciphertext, which was generated by algorithm $CipherG(Pub_{PEFKS-UE}, r_3, K)$. Specifically, the reduction proof in Table I proved that for any keyword $K \in \mathcal{K}$ with the parse $Fuz(K, \mathcal{K})||K_n$, the fuzzy keyword searchable ciphertext referring to K never leaks K_n under adaptive-ID and chosen plaintext attacks. On the other hand, to implement keyword guessing attack, any adversary fails to successfully guess K_n with the non-negligibly biased advantage, since the keyword space $\mathcal{K} = \sum^n$ has the uniform entropy such that each symbol of $\sum$ has the same or indistinguishable probability to be used. Conclusively, when the keyword space $\mathcal{K}$ has the uniform entropy and $|\mathcal{K}| \leq Poly(k)$, any adversary, who implements adaptive-ID and chosen plaintext attacks and keyword guessing attack, fails to successfully guess the exact keyword K with the non-negligibly biased advantage in the instance of PEFKS-UE.
- *The security of the universal transformation PEFKS-UE.* Without loss of generality, we assumed the universal anonymous IBE $UIBE$ has the Anon-ID-CPA security,

	Adversary A	Simulator/Challenger S	BF01 scheme IBE
Setup Phase	3) Receive $Pub_{PEFKS-UE}$.	2) Send $Pub_{PEFKS-UE} = \langle Pub_{IBE}, Fuz, H_1, \mathcal{K}\rangle$ to A.	1) Generate $Pub_{IBE} = \langle q, \mathbb{G}, \mathbb{G}_t, g, e, g_{pub}, H_1, H_2, \mathcal{M}\rangle$ and $Pri_{IBE} = s$ of BF01 scheme, then send Pub_{IBE} to S except H_2, which will be taken as a random oracle machine.
Query Phase	1) Query S for $\langle Or, X\rangle$, where $Or \in \{H_2, DTrapdoor\}$.	2) If $Or = DTrapdoor$, query IBE for $\langle Extract, Fuz(X, \mathcal{K})\rangle$ and $\langle Extract, X\rangle$, otherwise query IBE for $\langle Or, X\rangle$.	3) Respond queries of oracles H_2 and $Extract$.
	6) Receive the responses and execute the query many times.	5) Send the responses from IBE to A.	4) Send the responses to S.
Challenge Phase	1) Choose two different challenge keywords K^0 and K^1 under the following constraints: $\langle DTrapdoor, K^0\rangle$ and $\langle DTrapdoor, K^1\rangle$ were not queried; $Fuz(K^0, \mathcal{K}) = Fuz(K^1, \mathcal{K})$. Finally send $\langle K^0, K^1\rangle$ to S.	2) Randomly choose a plaintext $M \in \mathcal{M}$, then send $\langle M, K^0, K^1\rangle$ to IBE.	3) Randomly choose $b \xleftarrow{R} 0, 1$ and set $C_E^* = Encrypt(Pub_{IBE}, r_3, K^b, M)$.
	6) Receive the challenge ciphertext $\langle M, C_F^*, C_E^*\rangle$.	5) Send the challenge ciphertext $\langle M, C_F^*, C_E^*\rangle$ to A, where $C_F^* = Encrypt(Pub_{IBE}, r_3', ID_F^0, M)$ and $ID_F^0 = Fuz(K^0, \mathcal{K})$.	4) Send C_E^* to S.
Query Phase	(It is the same as the aforementioned query phase except querying S for $\langle Dtrapdoor, K^0\rangle$ and $\langle Dtrapdoor, K^1\rangle$.)	(It is the same as the aforementioned query phase.)	(It is the same as the aforementioned query phase.)
Final Phase	1) Send the guess b' on b to S.	2) Send the received b' to IBE.	3) Receive b'.

TABLE I

PROOF OF THE PREDICATION THAT THE INSTANCE OF PEFKS-UE TRANSFORMED FROM BF01 SCHEME INHERITS THE ANON-ID-CPA SECURITY.

	Adversary A	Simulator/Challenger S	Universal Anonymous IBE $UIBE$
Setup Phase	3) Receive $Pub_{PEFKS-UE}$.	2) Send $Pub_{PEFKS-UE} = \langle Pub_{UIBE}, Fuz, H_1, \mathcal{K}\rangle$ to A.	1) Generate Pub_{UIBE} and Pri_{UIBE} of $UIBE$, then send Pub_{UIBE} to S.
Query Phase	1) Query S for $\langle X\rangle$, where $X \in \mathcal{K}$.	2) Query $UIBE$ for $\langle H_1(Fuz(X, \mathcal{K}))\rangle$ and $\langle H_1(X)\rangle$.	3) Respond queries of oracle $Extract$.
	6) Receive the responses and execute the query many times.	5) Send the responses from $UIBE$ to A.	4) Send the responses to S.
Challenge Phase	1) Choose two different challenge keywords K^0 and K^1 under the following constraints: $\langle K^0\rangle$ and $\langle K^1\rangle$ were not queried by A; $Fuz(K^0, \mathcal{K}) = Fuz(K^1, \mathcal{K})$. Finally send the $\langle K^0, K^1\rangle$ to S.	2) Randomly choose a plaintext $M \in \mathcal{M}$, then send $\langle M, H_1(K^0), H_1(K^1)\rangle$ to $UIBE$.	3) Randomly choose $b \xleftarrow{R} 0, 1$, then set $C_E^* = Encrypt(Pub_{UIBE}, r_3, ID^b, M)$, where $ID^b = H_1(K^b)$.
	6) Receive the challenge ciphertext $\langle M, C_F^*, C_E^*\rangle$.	5) Send the challenge ciphertext $\langle M, C_F^*, C_E^*\rangle$ to A, where $C_F^* = Encrypt(Pub_{UIBE}, r_3', ID_F^0, M)$ and $ID_F^0 = H_1(Fuz(K^0, \mathcal{K}))$.	4) Send C_F^* to S.
Query Phase	(It is the same as the aforementioned query phase except querying S for $\langle K^0\rangle$ and $\langle K^1\rangle$.)	(It is the same as the aforementioned query phase.)	(It is the same as the aforementioned query phase.)
Final Phase	1) Send the guess b' on b to S.	2) Send the received b' to $UIBE$.	3) Receive b'.

TABLE II

PROOF OF THE PREDICATION THAT THE UNIVERSAL TRANSFORMATION PEFKS-UE INHERITS THE ANONYMITY.

since there is not any substantial difference among security proofs when the universal anonymous IBE $UIBE$ has the anonymity under the different attacks (e.g., chosen plaintext attacks and chosen ciphertext attacks [24]) . On one hand, we proved that the universal transformation PEFKS-UE inherits the anonymity of $UIBE$ by the reduction proof in Table II. Hence, the fuzzy keyword searchable ciphertext, generated by algorithm $CipherG$, keeps the privacy of K_n of any keyword $K = Fuz(K, \mathcal{K})||K_n$ under adaptive-ID and chosen plaintext attacks. On the other hand, since the keyword space $\mathcal{K}$ has the uniform entropy, any adversary, who implements the keyword guessing attack without the exact trapdoor of any keyword, fails to successfully guess the exact keyword with non-negligibly biased advantage even under the condition that $|\mathcal{K}| \leq Poly(k)$. Conclusively, when the keyword space has uniform entropy, the universal transformation PEFKS-UE, based on the universal anonymous IBE, keeps the privacy of K_n for any keyword $K = Fuz(K, \mathcal{K})||K_n$ under adaptive-ID and chosen plaintext attacks and keyword guessing attack, even with the condition that $|\mathcal{K}| \leq Poly(k)$.

In this section, under the conditions that the keyword space $\mathcal{K}$ has uniform entropy and $|\mathcal{K}| \leq Poly(k)$, we constructed a universal transformation from anonymous IBE to PEFKS-UE, then proposed an instance of PEFKS-UE based on the anonymous IBE scheme proposed by Boneh et al. [5], finally proved their security under adaptive-ID and chosen plaintext attacks and keyword guessing attack. In next section, we further weaken the condition such that the keyword space has non-uniform entropy and construct PEFKS-NE.

IV. PEFKS-NE

Let k be a security parameter. Let $\mathcal{E}_K$ bits denote the entropy of the keyword $K \in \mathcal{K}$, where $|\mathcal{K}| \leq Ploy(k)$. We partition all keywords into several subsets as follows:

1) Sort all keywords in ascending order of their entropy and index all keywords as $\{K^1, K^2, \ldots, K^{|\mathcal{K}|}\}$.
2) Partition $\{K^1, K^2, \ldots, K^{|\mathcal{K}|}\}$ as P, where $P =$

$$\begin{cases} \{\{K^1, K^2\}, \{K^3, K^4\}, \ldots, \{K^{|\mathcal{K}|-1}, K^{|\mathcal{K}|}\}\} \\ \quad if\ |\mathcal{K}|\ is\ even; \\ \{\{K^1, K^2\}, \ldots, \{K^{|\mathcal{K}|-4}, K^{|\mathcal{K}|-3}\}, \{K^{|\mathcal{K}|-2}, K^{|\mathcal{K}|-1}, K^{|\mathcal{K}|}\}\} \\ \quad if\ |\mathcal{K}|\ is\ odd. \end{cases} \tag{14}$$

Let $H_1 : \{0,1\}^* \rightarrow \{0,1\}^{k_2}$ be a collision resistance function. In PEFKS-NE, the function $Fuz(K^i, \mathcal{K})$ is adaptively redefined as follows:

1) If $|\mathcal{K}|$ is even, then return

$$\begin{cases} H_1(K^{i-1}||K^i) & if\ i\ is\ even; \\ H_1(K^i||K^{i+1}) & if\ i\ is\ odd. \end{cases} \tag{15}$$

2) Otherwise, return

$$\begin{cases} H_1(K^{|\mathcal{K}|-2}||K^{|\mathcal{K}|-1}||K^{|\mathcal{K}|}) & if\ i \geq |\mathcal{K}| - 2; \\ H_1(K^{i-1}||K^i) & if\ i\ is\ even; \\ H_1(K^i||K^{i+1}) & if\ i\ is\ odd. \end{cases} \tag{16}$$

Referring to the function Fuz, we can easily verify following properties:

- For any subset in P, the keywords belonging to the same subset have the same output of Fuz. (Formally Speaking, for any subset $\{K^i, K^{i+1}\} \in P$, $Fuz(K^i, \mathcal{K}) = Fuz(K^{i+1}, \mathcal{K})$ holds.)
- Furthermore, for any two different subsets in P, they have different outputs of Fuz. (Formally speaking, for any two subsets $\{K^i, K^{i+1}\}$ and $\{K^j, K^{j+1}\}P$, where $i \neq j$, $Fuz(K^i, \mathcal{K}) \neq Fuz(K^j, \mathcal{K})$ holds as the collision resistance of H_1.)

We construct a universal transformation from anonymous IBE to PEFKS-NE in next subsection.

A. A Universal Transformation

Let $UIBE = (Setup, Extract, Encrypt, Decrypt)$ be a universal anonymous IBE defined in Definition 1. Let $H_2 : \{0,1\}^{k_2} \bigvee \mathcal{K} \rightarrow \mathcal{ID}$ be a collision resistant function. PEFKS-NE consists of the following algorithms:

- $SysG(k, r_1)$: Take as input a security parameter k and a random tape r_1, then run algorithm $Setup(k, r_1)$ of $UIBE$ to generate a pair of public-and-private system parameters that

$$\begin{aligned} Pub_{PEFKS-NE} &= \langle Pub_{UIBE}, Fuz, H_1, H_2, \mathcal{K} \rangle \\ Pri_{PEFKS-NE} &= Pri_{UIBE} \end{aligned} \tag{17}$$

- $DTrapdoor(Pri_{PEFKS-NE}, r_2, r_2', K^i)$: Take as input a private system parameter $Pri_{PEFKS-NE}$, two random tapes r_2 and r_2', and a keyword $K^i \in \mathcal{K}$, then generate a fuzzy search trapdoor FT_{K^i} and an exact search trapdoor ET_{K^i}, where

$$\begin{aligned} FT_{K^i} &= Extract(Pri_{UIBE}, r_2', H_2(Fuz(K^i, \mathcal{K})) \\ ET_{K^i} &= Extract(Pri_{UIBE}, r_2, H_2(K^i)) \end{aligned} \tag{18}$$

- $CipherG(Pub_{PEFKS-NE}, r_3, r_3', K^i)$: Take as input a public system parameters $Pub_{PEFKS-NE}$, two random tapes r_3 and r_3' and a keyword $K^i \in \mathcal{K}$, then randomly choose a message $M \in \mathcal{M}$, finally generate a fuzzy keyword searchable ciphertext $\langle M, C_F, C_E \rangle$, where

$$\begin{aligned} C_F &= Encrypt(Pub_{UIBE}, r_3, H_2(Fuz(K^i, \mathcal{K})), M) \\ C_E &= Encrypt(Pub_{UIBE}, r_3', H_2(K^i), M) \end{aligned} \tag{19}$$

- $FuzzTest(Pub_{PEFKS-NE}, FT_{K^i}, \langle M, C_F \rangle)$: Take as input a public system parameters $Pub_{PEFKS-NE}$, a fuzzy search trapdoor FT_{K^i} and a searchable ciphertext $\langle M, C_F \rangle$, then return B, where

$$B = \begin{cases} 1 & if\ M = Decrypt(Pub_{UIBE}, FT_{K^i}, C_F); \\ 0 & otherwise. \end{cases} \tag{20}$$

- $ExactTest(Pub_{PEFKS-NE}, ET_{K^i}, \langle M, C_E \rangle)$: Take as input a public system parameters $Pub_{PEFKS-NE}$, an

exact search trapdoor ET_{K^i} and a searchable ciphertext $\langle M, C_E\rangle$, then return B, where

$$B = \begin{cases} 1 & if\ M = Decrypt(Pub_{UIBE}, ET_{K^i}, C_E); \\ 0 & otherwise. \end{cases} \tag{21}$$

The consistency of PEFKS-NE. According to Theorem 4.2 in [20], we easily find that PEFKS-NE has the consistence, when the anonymous IBE scheme $UIBE$ satisfies the semantic security.

B. Security Proofs

Without loss of generality, we also assume the universal anonymous IBE $UIBE$ satisfied the Anon-ID-CPA security. On one hand, the reduction proof in Table III proved that PEFKS-NE inherits the Anon-ID-CPA security of $UIBE$. Hence, for any keyword $K^i \in \mathcal{K}$ used to generate a fuzzy keyword searchable ciphertext $\langle M, C_E, C_F\rangle$ by algorithm $CipherG$, no adversary knows K^i from C_F under adaptive-ID and chosen plaintext attacks.

On the other hand, no adversary knows the exact trapdoor of any keyword in PEFKS-NE. Hence, they fail to deterministically guess the exact keyword by implementing keyword guessing attack even under the condition that $|\mathcal{K} \leq Poly(k)|$. However, in PEFKS-NE, an adversary may have the non-negligibly biased advantage to successfully guess the exact keyword. Moreover, we believe that the non-negligibly biased advantage can not be absolutely avoided.

Specifically, without loss of generality we assume an adversary known the fuzzy search trapdoor $FT_{K^i/K^{i+1}}$ of the subset $\{K^i, K^{i+1}\} \in P$. To implement keyword guessing attack on PEFKS-NE, the adversary can efficiently find out the subset $\{K^i, K^{i+1}\}$. Furthermore, according to the partition of $\mathcal{K}$ at the beginning of this section, he has a biased advantage $|2^{-\mathcal{E}_{K^i}} - 2^{-\mathcal{E}_{K^{i+1}}}|$ to decide which one between K^i and K^{i+1} has the higher probability to be used to generate $FT_{K^i/K^{i+1}}$. The further analysis should be divided into the following two cases:

- Case 1: $|2^{-\mathcal{E}_{K^i}} - 2^{-\mathcal{E}_{K^{i+1}}}| < \frac{1}{Poly(k)}$. In this case, the adversary has the negligibly biased advantage. Hence, he fails to efficiently decide which one has the higher probability to be used. Formally speaking, let $Pr(K^i|FT_{K^i})$ denote the posterior probability of the event that K^i is employed under the condition that FT_{K^i} is given. We have $Pr(K^i|FT_{K^i}) \approx Pr(K^{i+1}|FT_{K^i})$. It means that the keywords K^i and K^{i+1} almost have the maximum "posterior entropy".
- Case 2: $|2^{-\mathcal{E}_{K^i}} - 2^{-\mathcal{E}_{K^{i+1}}}| \geq \frac{1}{Poly(k)}$. In this case, the adversary has the non-negligibly biased advantage. However, we have reduced the advantage as much as possible. Moreover, we justify that the biased advantage can not be avoid. Giving an extreme example, $\mathcal{K} = \{K^1, K^2\}$ and $\mathcal{E}_{K^1} \ll \mathcal{E}_{K^2}$. It induces that the adversary has enough confidence to decide that K^1 was used to generate the searchable ciphertext, even without considering the cryptosystem you employed. Consequently, since the biased advantage is a natural character of keywords, nobody can absolutely avoid it. Fortunately we have reduced the biased advantage as much as possible.

C. Insecurity of the Further Reduction

At the end of Subsection IV-B, we stated that we have reduced the biased advantage as much as possible. For justifying it, we will propose a method, which is the only one method that can further reduce the biased advantage, then illuminate the insecurity of the method.

Referring to the partition of $\mathcal{K}$ at the beginning of this section, we sorted $\mathcal{K}$ in ascending order of keyword entropy and sequentially partition keywords. Hence, in the partition phase, we did not consider the difference between the entropy of neighboring keywords. However, when considering it, this method is provably insecure. For example, let $\mathcal{K} = \{K^1, K^2, K^3, K^4\}$ be an sorted keywords in ascending order of their entropy. We combine a chosen keyword with one of its neighboring keywords, according to which neighboring keywords has the minimum entropy difference between the chosen keyword and the neighboring keyword. Hence, assuming $|\mathcal{E}_{K^2} - \mathcal{E}_{K^3}| < |\mathcal{E}_{K^1} - \mathcal{E}_{K^2}|$, we result the partition $\{\{K^1, K^2\}, \{K^2, K^3\}, \{K^3, K^4\}\}$. Consequently, if an adversary has the fuzzy search trapdoor FT_{K^1/K^2}, he can easily find out $\{K^1, K^2\}$ by implementing keyword guessing attack on PEFKS. In addition, he can deterministically guess K^1 rather than with any biased advantage. Since assuming the fuzzy search trapdoor FT_{k^1/K^2} was generated from K^2, the adversary should find out $\{K^2, K^3\}$ rather than $\{K^1, K^2\}$ according to $|\mathcal{E}_{K^2} - \mathcal{E}_{K^3}| < |\mathcal{E}_{K^1} - \mathcal{E}_{K^2}|$. Hence, if the adversary find out $\{K^1, K^2\}$, he can deterministically decide that the fuzzy search trapdoor was generated from K^1 rather then K^2.

V. Conclusion and Future Work

In PEKS, the sever, which received search trapdoors from the receiver to search the public-key encrypted keywords, can deterministically know the content of keywords by implementing keyword guessing attack. Moreover, this attack is efficient under the practical condition that the size of the keyword space is not more than the polynomial level. In order to withstand keyword guessing attack, we first proposed a novel concept that public-key encryption with fuzzy keyword search. Furthermore, under two different conditions that keyword space have uniform and non-uniform entropy, we presented two universal transformations from anonymous IBE to PEFKS, which were respectively called PEFKS-UE and PEFKS-NE. Their provable securities are also given under adaptiv-ID and chosen plaintext attacks and keyword guessing attack. Although, in PEFKS-NE we failed to absolutely avoid the biased advantage in keyword guessing attack, we have reduced the advantage as much as possible. Moreover, we proved that the biased advantage is caused by the natural character of keywords and no cryptosystem can absolutely avoid it. Frankly speaking, merely considering the security, our solution is not perfect. However, it is presented as the state of the art and absolutely securer than PEKS in practice.

	Adversary A	Simulator/Challenger S	Universal Anonymous IBE $UIBE$
Setup Phase	3) Receive $Pub_{PEFKS-NE}$.	2) Send $Pub_{PEFKS-NE} = \langle Pub_{UIBE}, Fuz, H_1, H_2, \mathcal{K}\rangle$ to A.	1) Generate Pub_{UIBE} and Pri_{UIBE} of $UIBE$, then send Pub_{UIBE} to S.
Query Phase	1) Query S for $\langle K^i\rangle$, where $K^i \in \mathcal{K}$.	2) Query $UIBE$ for $\langle Extract, H_2(Fuz(K^i, \mathcal{K}))\rangle$ and $\langle Extract, H_2(K^i)\rangle$.	3) Respond queries of oracle $Extract$.
	6) Receive the responses and execute the query many times.	5) Send the responses from $UIBE$ to A.	4) Send the responses to S.
Challenge Phase	1) Choose two different challenge keywords K^i and K^j (assuming $i < j$ without loss of generality) under the following constraints: $\langle K^i\rangle$ and $\langle K^j\rangle$ were not queried by A; $Fuz(K^i, \mathcal{K}) = Fuz(K^j, \mathcal{K})$. Finally send $\langle K^i, K^j\rangle$ to S.	2) Choose a plaintext $M \in \mathcal{M}$ and send $\langle M, ID^0, ID^1\rangle$ to $UIBE$, where $ID^0 = H_2(K^i)$ and $ID^1 = H_2(K^j)$.	3) Randomly choose $b \overset{R}{\leftarrow} 0, 1$ and set $C^*_E = Encrypt(Pub_{UIBE}, r_3, ID^b, M)$.
	6) Receive the challenge ciphertext $\langle M, C^*_F, C^*_E\rangle$.	5) Send the challenge ciphertext $\langle M, C^*_F, C^*_E\rangle$ to A, where $C^*_F = Encrypt(Pub_{UIBE}, r'_3, ID^0_F, M)$, where $ID^0_F = H_2(Fuz(K^i, \mathcal{K}))$.	4) Send C^*_E to S.
Query Phase	(It is the same as the aforementioned query phase except querying S for $\langle K^i\rangle$ and $\langle K^j\rangle$.)	(It is the same as the aforementioned query phase.)	(It is the same as the aforementioned query phase.)
Final Phase	1) Send the guess i' on $\{i, j\}$ to S.	2) Send $b' = 0$ to $UIBE$ if $i' = i$, otherwise send $b' = 1$ to $UIBE$.	3) Receive the guess b'.

TABLE III
PROOF OF THE PREDICATION THAT THE UNIVERSAL TRANSFORMATION PEFKS-NE INHERITS THE ANONYMITY.

Both in PEKS and PEFKS, the encrypted keywords are searched one by one. Hence it is a crucial performance problem that how to build the index for these ciphertexts having the same keyword without leaking the content. In this way, the ciphertexts generated from the same keyword can be searched only one step. Although this problem is easy to be solved in the RO model, our future challenge is to solve it in the standard model.

REFERENCES

[1] D. Boneh, G. D. Crescenzo, and R. O. et al., "Public key encyrption with keyword search," in *Advances in Cryptology-EUROCRYPT 2004*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Springer-Verlag, 2004, pp. 506–522.

[2] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, Berkeley, CA , USA, 2000, pp. 44–55.

[3] R. Agrawal, J. Kiernan, and R. S. et al., "Order preserving encryption for numeric data," in *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. Paris, France: ACM, 2004, pp. 563–574.

[4] R. Curtmola, J. Garay, and S. K. et al., "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proceedings of the 13th ACM conference on Computer and communications security*. ACM, 2006, pp. 79–88.

[5] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology-CRYPTO 2001*, ser. LNCS, J. Kilian, Ed., vol. 2139. Santa Barbara, California, United States: Springer-Verlag, 2001, pp. 213–239.

[6] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *Advances in Cryptology-CRYPTO 2006*, ser. LNCS, C. Dwork, Ed., vol. 4117. Santa Barbara, California, United States: Springer-Verlag, 2006, pp. 290–307.

[7] C. Gentry, "Practical identity-based encyrption without random oracles," in *Advances in Cryptology-EUROCRYPT 2006*, ser. LNCS, S. Vaudenay, Ed., vol. 4004. Russia: Springer-Verlag, 2006, pp. 445–464.

[8] L. Ducas, "Anonymity from asymmetry: New constructions for anonymous hibe," in *The Cryptographers' Track at the RSA Conference 2010*, ser. LNCS, J. Pieprzyk, Ed., vol. 5985. San Francisco, CA, USA: Springer Berlin, 2010, pp. 148–164.

[9] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in *WISA 2004*, ser. LNCS, C. Lim and M. Yung, Eds., vol. 3325. Spring-Verlag, 2004, pp. 73–86.

[10] Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," in *Pairing 2007*, ser. LNCS, T. Takagi, Ed., vol. 4575. Springer-Verlag, 2007, pp. 2–22.

[11] J. Bethencourt, T.-H. H. Chan, and A. P. et al., "Anonymous multi-attribute encryption with range query and conditional decryption," Carnegie Mellon University, Tech. Rep. CMU-CS-06-135, 2006.

[12] E. Shi, J. Bethencourt, and T.-H. H. C. et al., "Multi-dimensional range query over encrypted data," Carnegie Mellon University, Tech. Rep. CMU-CS-06-135, 2007.

[13] D. Boneh and B. Waters, "conjunctive, subset, and range

queries on ecrypted data," in *proceedings of TCC'07*, ser. LNCS, S. P. Vadhan, Ed., vol. 4392. Springer-Verlag, 2007, pp. 535–554.

[14] J. Camenisch, M. Kohlweiss, and A. R. et al., "Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data," in *Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC '09*, ser. LNCS, vol. 5443. CA: Springer-Verlag, 2009, pp. 196–214.

[15] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," in *ICICS 2005*, ser. LNCS, S. et al., Ed., vol. 3783. Springer-Verlag, 2005, pp. 414–426.

[16] E.-K. Ryu and T. Takagi, "Efficient conjunctive keyword-searchable encryption," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*. Niagara Falls, Ontario, Canada: IEEE Computer Society, 2007, pp. 409 – 414.

[17] W. Harrower, "Searching encrypted data," Department of Computing, Imperial College London, Tech. Rep., 2009.

[18] I. R. Jeong, J. O. Kwon, and D. H. et al., "Constructing peks schemes secure against keyword guessing attacks is possible?" *Computer Communications*, vol. 32, no. 2, pp. 394–396, 2009.

[19] J. W. Byun, H. S. Rhee, and H.-A. P. et al., "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," in *Secure Data Management*, ser. LNCS, W. Jonker and M. Petkovic, Eds., vol. 4165. Springer-Verlag, 2006, pp. 75–83.

[20] M. Abdalla, M. Bellare, and D. C. et al., "Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," in *Advances in Cryptology-CRYPTO 2005*, ser. LNCS, V. Shoup, Ed., vol. 3621. Santa Barbara, California, United States: Springer-Verlag, 2005, pp. 205–222.

[21] A. J. Menezes, T. Okamoto, and S. A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. 39, no. 5, pp. 1639–1646, 1993.

[22] G. Frey, M. Muller, and H.-G. Ruck, "The tate pairing and the discrete logarithm applied to elliptic curve cryptosystems," *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. 45, no. 5, pp. 1717–1719, 1999.

[23] M. Bellare, A. Boldyreva, and A. D. et al., "Key-privacy in public-key encryption," in *Advances in Cryptology-ASIACRYPT 2001*, ser. LNCS, C. Boyd, Ed., vol. 2248. Australia: Springer-Verlag, 2001, pp. 566–582.

[24] X. Boyen, Q. Mei, and B. Waters, "Direct chosen ciphertext security from identity-based techniques," in *the proceedings of the 12th ACM Conference on Computer and Communications Security-CCS 2005*. Alexandria, VA: ACM Press, 2005, pp. 320–329.