Fast Keyword Search over Encrypted Data with Short Ciphertext in Clouds

Yi-Fan Tseng[a], Chun-I Fan[b,c,d,*], Zi-Cheng Liu[b]

[a] *Department of Computer Science, National Chengchi University, Taipei, Taiwan.*
[b] *Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan*
[c] *Information Security Research Center, National Sun Yat-sen University, Kaohsiung, Taiwan*
[d] *Intelligent Electronic Commerce Research Center, National Sun Yat-sen University, Kaohsiung, Taiwan*

Abstract

Nowadays, it is convenient for people to store their data on clouds. To protect the privacy, people tend to encrypt their data before uploading them to clouds. Due to the widespread use of cloud services, public key searchable encryption is necessary for users to search the encrypted files efficiently and correctly. However, the existing public key searchable encryption schemes supporting monotonic queries suffer from either infeasibility in keyword testing or inefficiency such as heavy computing cost of testing, large size of ciphertext or trapdoor, and so on. In this work, we first propose a novel and efficient anonymous key-policy attribute-based encryption (KP-ABE). Then by applying Shen *et al.*'s generic construction to the proposed anonymous KP-ABE, we obtain an efficient and expressive public key searchable encryption, which to the best of our knowledge achieves the best performance in testing among the existing such schemes. Only 2 pairings are needed in testing. By applying our searchable encryption, one is able to expressively and efficiently search their encrypted data on clouds, without leaking the keyword information. Besides, we also implement our scheme and others with Python for comparing the performance. From

*Corresponding author
Email addresses: yftseng@cs.nccu.edu.tw (Yi-Fan Tseng), cifan@mail.cse.nsysu.edu.tw (Chun-I Fan), tcdiadem@gmail.com (Zi-Cheng Liu)

the implementation results, our scheme owns the best performance on testing, and the size of ciphertexts and trapdoors are smaller than most of the existing schemes.
Keywords: Public Key Searchable Encryption, Key-Policy Attribute-Based Encryption, Anonymous KP-ABE, The Standard Model, Monotonic Access Structure

1. Introduction

Cloud technology has been thriving around the world recently, which allows data users to access unlimited storage resources. Users are able to access their data anytime and anywhere. By remotely accessing data on clouds, the cost of data storage and management has been greatly reduced. People tend to store their data on clouds so that they can back up the data and retrieve them anytime and anywhere. Due to the advantage of cloud technology, the security issues on clouds have also been noticed [1], and cryptographic primitives [2, 3, 4] are found useful in dealing with these security issues, e.g. the cloud service provider may be curious about the sensitive data stored on the clouds, even profit-driven to leak the data. Consider the following scenario. In a company, employees are asked to store the commercial documents in the company's private cloud. In order to prevent unauthorized access, it is necessary to store the documents in encrypted form. Besides, the documents may come from customers, and thus they should be transmitted in encrypted form. This is a common business model of nowadays. In such scenario, it is significant for the employees to efficiently and securely search the required encrypted files. A practical solution to this problem is to apply searchable encryption.

In 2000, Song et al. [5] first gave the definition of searchable encryption (SE). In an SE scheme, a data owner can encrypt keywords and upload it with the encrypted data so that users can find the desired data by searching the encrypted keywords. In 2004, Boneh et al. [6] first proposed a public key encryption with

keyword search (PEKS) (a.k.a. public key searchable encryption). They combined the public key setting and the keyword search encryption, and discussed the relationship between PEKS and identity-based encryption (IBE). Note that public key searchable encryption is different from private key searchable encryption (a.k.a. searchable symmetric encryption) [7]. The former belongs to the family of public key primitives, where an encryptor is allowed to be different to the owner of private key, while in a private key searchable encryption, the one who encrypts the data must be the same as the one who is able to decrypt the data. In this manuscript, we focus on solving the emerging problems in the realm of PEKS. Following Boneh's pioneering work, Abdalla et al. [8] proposed a generic construction of PEKS from anonymous IBE, where an encryption reveals nothing about its receiver. However, [8, 6] only support equality queries. It is necessary to construct an PEKS scheme with more expressive queries such as conjunction, disjunction and monotonic formulas, which supporting both AND/OR gates in a boolean formula, to make the search more accurate and flexible. In 2007, Boneh et al. [9] proposed a searchable encryption scheme supporting conjunctive, subset, and range queries in public key setting. In the next year, Katz et al. [10] first introduced inner-product predicate encryption (IPE) [11, 12] which can be extended to PEKS supporting disjunctive queries. Nevertheless, it is inefficient in this way because the size of the ciphertext and the search token would superpolynomially blow up [13]. Another shortcoming of Katz et al.'s work is that, their scheme is constructed under composite-order bilinear groups whose performance is notoriously worse than prime-order bilinear groups. According to [14], the length of a group element in a composite-order group is 12 times larger than that in a prime-order group. Besides, the bilinear pairings in composite-order groups are 254 times slower than those in prime-order groups for the same 128-bit security. In 2018, there are several related works [15], [16],[17],[18] that have been proposed. In these schemes, the authors explored keyword search in attribute-based encryption so that the scheme can support access control and keyword search simultaneously. However, these schemes cannot support expressive search queries. They only support searching

on single keyword. To achieve expressive queries (i.e. conjunction/disjunction of keywords) is significant for cloud applications due to its convenience. Fig 1 shows an example for PEKS supporting expressive queries. As a public-key primitive, any user can upload an encrypted file tagged with some keywords to a cloud service provider. Besides, any user is also allowed to make queries of the conjunction/disjunction for some keywords to search the files they want. Once the cloud service provider receives a search query from a user, it will use the **Test** algorithm to find the related ciphertexts, which will be returned to the user.

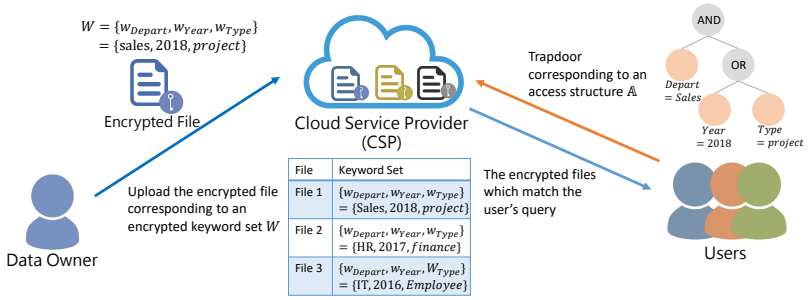


Figure 1: Example for the Application of PEKS Supporting Expressive Queries

To achieve more expressive queries, Lai et al. [13] proposed a PEKS scheme motivated from Lewko et al.'s [19] key-policy attribute-based encryption (KP-ABE) in 2013. In 2012 Han et al. [20, 21] proposed a generic construction for attribute-based encryption with keyword search (ABEKS). In 2020, Shen *et al.* [22] further gave a generic construction for building PEKS from anonymous KP-ABE[1]. Note that the notion of ABEKS is different from PEKS. The former needs a trusted third party for issuing attribute keys to users, while in a PEKS scheme, users generate their public/secret key by themselves. However, there exists a common problem of these schemes [13, 21, 22, 23]. The test algorithm in these schemes will not be conducted successfully. For instance, the test

[1]For more discussion on anonymous KP-ABE, please refer to Section Appendix A.

algorithm in [13] needs to compute the following formula.

$$\hat{C} = \prod_{i \in \mathcal{I}} \left(\frac{e(C_0, K_{1,i})}{e(C_{\rho(i)}, K_{2,i})} \right)^{\omega_i} \tag{1}$$

In formula 1, $K_{1,i}, K_{2,i}$ represent the search token associated with keyword $\rho(i)$ and $C_{\rho(i)}$ represents the ciphertext component associated with keyword $\rho(i)$, where ρ is a map from indices to keywords. We can observe that it is necessary to know the correlation between $(K_{1,i}, K_{2,i})$ and $C_{\rho(i)}$, which implies that the test algorithm needs to know the corresponding keyword of $C_{\rho(i)}$. If the underlying KP-ABE is anonymous, it will be infeasible in conducting test algorithm. The test algorithms in [20, 21, 22, 23, 24] are similar to that in [13], and thus they suffer from the same problem.

In order to solve the correlation problem, in 2018, Cui et al. [25] proposed a PEKS scheme with weaker anonymity notion. They separate a keyword into a keyword name and a keyword value. For instance, in the case of ("gender" = "female"), "gender" is the keyword name and "female" is the keyword value. The weaker anonymity in [25] only guarantees that a ciphertext reveals nothing on its keyword values, while the keyword names are attached to the ciphertext. In the same year, Meng et al. [26] improved the efficiency of [25] by aggregating the ciphertext components for each attribute into a group element. However, their Test algorithm requires that all attributes in a ciphertext should appear in the access structure, or it would fail. Besides, there exists a common problem in [25, 26]. That is their schemes need an online and trusted third party to generate search tokens which is an unreasonable assumption in cryptography.

1.1. Contribution

In this work, we aim at proposing an efficient PEKS supporting expressive search queries. Due to [22], we have a new approach to build a PEKS scheme. Therefore, we first propose a novel anonymous KP-ABE with provably security. Then, by adopting the generic construction shown in [22], we obtain a novel

PEKS from KP-ABE supporting monotonic access structure with the following advantages.

1. **Expressive queries:** The proposed scheme supports monotonic formula in search queries.
2. **High efficiency:** The proposed scheme is constructed under prime-order bilinear groups. Moreover, the pairings performed in the Test algorithm is independent of the number of attributes in ciphertexts and search tokens. Besides the length of ciphertexts in the proposed scheme is shorter than most of the existing schemes.
3. **Formal security proof:** The proposed scheme is proven to be fully secure in the standard model.

When we express an access structure by a boolean formula, by "monotonic access structure", we means that both AND/OR gates is allowed to be used in the boolean formula.

1.2. Related Works

In this section, we briefly introduce some works [13, 21, 22, 23, 25, 26, 27] aiming at solving the similar problem, or trying to achieve similar goal to us. The papers [13, 22, 23, 25, 26] both give PEKS supporting expressive queries, while [21] gives a ABEKS scheme supporting expressive queries. As we mentioned before, PEKS and ABEKS are different notions. Specifically, the paradigm of our work is similar to that in [22], i.e., designing a anonymous KP-ABE and then obtaining a PEKS via generic transformation. However, the feasibility of Test algorithm in [13, 21, 22, 23] is problematic. In those schemes, to perform Test algorithm, the cloud service provider needs to successfully pair the ciphertext component to the corresponding search token component. However, this needs the information of keywords in the ciphertext, while such information should be hidden due to the security requirements of searchable encryption. Though [25, 26] solve the feasibility issue by splitting a keyword into a keyword name and a keyword value, these schemes needs a trusted third party for generating search

tokens. Since users would make data search queries anytime and anywhere, the trusted third party should be online, which will cause a great burden to the system. There is another recent paper [27] worthy of mentioning. Though the authors of [27] does not propose a PEKS, they propose an anonymous KP-ABE. Thus, we can adopt a transformation on their KP-ABE scheme to obtain a new PEKS. However, the Test algorithm of our scheme is much more efficient then the PEKS obtained via [27]' scheme. The details of the comparison between our scheme and these schemes are demonstrated in Section 5.

1.3. Organization

The following of this manuscript is organized as follows. In Section 2, we show the preliminary knowledge, including the definition of PEKS, KP-ABE, access structure, and the complexity assumption the security of our scheme bases on. Besides, Shen *et al.*'s transformation from anonymous KP-ABE to PEKS is also introduced in Section 2. In Section 3, we demonstrate the proposed anonymous KP-ABE scheme, and give a simple example to explain how our KP-ABE can be applied to Shen *et al.*'s transformation for constructing an expressive and efficient PEKS. In Section 4, show the security proofs for the security of the proposed KP-ABE and PEKS. The performance evaluation and property comparison to other existing works is given in Section 5. Finally, we conclude our work in Section 6, and give some brief introduction to anonymous KP-ABE in Appendix A.

2. Preliminaries

In this section, we introduce the related formal definitions of public key searchable encryption and other preliminaries.

2.1. Bilinear Mapping

In this subsection, we will introduce the definition of bilinear mappings.

Definition 2.1. Let $\mathbb{G}, \hat{\mathbb{G}}$ and $\mathbb{G}_T$ be all multiplicative cyclic groups of prime order p.

A bilinear mapping $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ satisfies the following properties in which $g, \hat{g}$ are generators of $\mathbb{G}, \hat{\mathbb{G}}$, respectively.

- **Bilinearity:** $e(h^a, \hat{h}^b) = e(h, \hat{h})^{ab}$, $\forall h \in \mathbb{G}, \hat{h} \in \hat{\mathbb{G}}$ and $a, b \in \mathbb{Z}_p$.
- **Non-Degeneracy:** There exist $h \in \mathbb{G}$ and $\hat{h} \in \hat{\mathbb{G}}$ such that $e(h, \hat{h}) \neq 1$.
- **Computability:** There exists an efficient algorithm to compute $e(h, \hat{h}), \forall h \in \mathbb{G}, \hat{h} \in \hat{\mathbb{G}}$.

2.2. The DBDH-3 Problem

In this subsection, we will introduce the definition of the DBDH-3 problem shown in [28], which is a variant of the decisional bilinear Diffie-Hellman in asymmetric paring groups.

Definition 2.2. Given $(g, \hat{g}, g^a, g^b, g^c, \hat{g}^a, \hat{g}^b, Y)$, where $a, b, c \xleftarrow{\$} \mathbb{Z}_p$, decide whether $Y = e(g, \hat{g})^{abc}$ or a random element in $\mathbb{G}_T$. Let $D = (g, \hat{g}, g^a, g^b, g^c, \hat{g}^a, \hat{g}^b)$. We say that an algorithm $\mathcal{B}$ that outputs a bit has the advantage ϵ in solving the DBDH problem if

$$\left| \Pr[\mathcal{B}(D, Y = e(g, \hat{g})^{abc}) = 1] - \Pr[\mathcal{B}(D, Y \xleftarrow{\$} \mathbb{G}_T) = 1] \right| \geq \epsilon.$$

2.3. Linear Secret-Sharing Scheme (LSSS)

We adapt the definition from those given in [29].

Definition 2.3. [29] A secret-sharing scheme $\prod$ over a set of parties P is called linear (over $\mathbb{Z}_p$) if

1. the shares for each party form a vector over $\mathbb{Z}_p$.
2. There exists a matrix $\mathbb{M}$ with ℓ rows and n columns called the share-generation matrix for $\prod$, which can be computed from the access structure $\mathbb{A}$ of attribute names. For the i-th row of $\mathbb{M}$, $i = 1, \ldots, \ell$, we let the function ρ define the party labelling row i as $\rho(i)$ which maps the row i to

an attribute name. We consider a column vector $\vec{v} = (s, r_2, \ldots, r_n)$, where s is the value to be shared, and $r_2, ..., r_n \in \mathbb{Z}_p$ are randomly chosen, and thus $\mathbb{M}\vec{v}$ is the vector of ℓ shares of the value s according to $\prod$. The share $\lambda_i = \mathbb{M}_i \vec{v}^\top$ belongs to party $\rho(i)$, where $\mathbb{M}_i$ is the i-th row of $\mathbb{M}$.

According to [29], every linear secret sharing-scheme satisfying the above definitions also enjoys the linear reconstruction property. Let S be an authorized set and $I = \{i : \rho(i) \in S\} \subseteq \{1, 2, \ldots, \ell\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$, such that $\sum_{i \in I} \omega_i \lambda_i = s$.

2.4. Access Structure

In this subsection, we will introduce the definition of access structures used in the proposed scheme.

Definition 2.4. An access structure $\mathbb{A}$ in our scheme contains an $(\mathbb{M}, \rho)$ corresponding to attribute names and a set $L = (z_{\rho(1)}, \ldots, z_{\rho(\ell)})$ corresponding to attribute values of $\rho(1), \ldots, \rho(\ell)$, respectively. Given an access structure $\mathbb{A} = (\mathbb{M}, \rho, L)$ a set $S = (v_1, \ldots, v_t)$ corresponding to the values of attribute names $1, \ldots, t$, respectively, we say that S satisfies $\mathbb{A}$ (denoted by $S \preceq \mathbb{A}$), if:

- there exists an index set $I = \{i : \rho(i) \in [1, t]\}$, such that, there exsit constants $\{\omega_i\}_{i \in I}$ satisfing $\sum_{i \in I} \omega_i \mathbb{M}_i = (1, 0, \ldots, 0)$;
- for $i \in I$, $v_{\rho(i)} = z_{\rho(i)}$.

2.5. Public Key Searchable Encryption

A public key searchable encryption [9, 13] consists of four algorithms.

- **Setup**(1^λ) : Take as input a security parameter λ. It outputs a public/secret key pair (PK, SK).
- **Encrypt**(PK, W) : Take as inputs the public key PK and a keyword set W. It outputs a ciphertext CT_W.

- **Trapdoor**$(PK, SK, \mathcal{P})$: Take as inputs the public key PK, the secret key SK and a predicate $\mathcal{P}$. It outputs a search token $TK_{\mathcal{P}}$. Note that the search tokens are also known as trapdoors in the literatures. In this work, we sometimes use "trapdoor" to denote a search token.

- **Test**$(PK, TK_{\mathcal{P}}, CT_W)$: Take as inputs the public key, a search token $TK_{\mathcal{P}}$ and a ciphertext CT_W. If the keyword set W satisfies the predicate $\mathcal{P}$, the algorithm outputs 1; otherwise, outputs 0.

The predicate supported by our scheme is monotonic formula represented by linear secret sharing schemes. Next we show the definition for public key searchable encryption, called IND-CKA security (i.e. indistinguishability against chosen keyword attacks). The notion states that a ciphertext in an SE scheme reveals no information about its keywords.

Definition 2.5. (IND-CKA Security)

- Setup: A challenger runs the **Setup** algorithm and gives the public parameters to the adversary.

- Phase 1: The adversary is allowed to issue polynomially many queries for trapdoors T with access structures $\mathbb{A}_j$'s.

- Challenge: The adversary submits two equal-size keywords sets W_0^* and W_1^*. The sets W_0^* and W_1^* should not satisfy any trapdoor that has been queried in Phase 1. The challenger flips a random coin b, and generate a ciphertext C^* with W_b^*. The ciphertext C^* is passed to the adversary.

- Phase 2: The adversary repeats the steps in Phase 1.

- Guess: The adversary outputs the guess $b' \in \{0, 1\}$ of b and wins the game if $b' = b$.

The advantage of the adversary in this game is defined as $Adv_{\mathcal{A}}^{IND\text{-}CKA} = |Pr[b' = b] - \frac{1}{2}|$. A PEKS scheme is said to be semantically secure if for every polynomial-time adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{IND\text{-}CKA}$ is at most negligible.

Remark 1. There is another security notion called "keyword privacy", which is analogous to the notion "function-private" [30] in functional encryption. Keyword privacy states that a trapdoor reveals no information about its keywords (or policy). However, as that stated in [24], it is impossible to achieve for PEKS. The reason is obvious, i.e. given a trapdoor, anyone can generate a ciphertext for any keywords under her/his choice, and thus reveal the information of the given trapdoor by performing the Test algorithm with the trapdoor and the ciphertext. Since our goal is to solve the problems for PEKS, we only focus on the IND-CKA security of the proposed scheme.

2.6. Definition and Security Model for Key-Policy Attribute-Based Encryption

KP-ABE was first proposed by Goyal et al. in [31], which is the dual construction of ciphertext-policy attribute-based encryption (CP-ABE) [32, 33]. A KP-ABE consists of the following four algorithms.

- **Setup**(1^λ) : The input is the security parameter 1^λ. The algorithm outputs the public parameter PK and the master secret key MSK.
- **KeyGen**$(PK, MSK, \mathbb{A})$: The inputs are the public parameter PK, master secret key MSK, and the access structure $\mathbb{A}$ which is assigned by Key Generation Center (KGC) to the user. The algorithm outputs a decryption key $SK_{\mathbb{A}}$ which contains the information of access structure.
- **Encrypt**(PK, S, M) : The inputs are the public parameter $param$, a set of descriptive attributes S, and a message M. The algorithm outputs a ciphertext CT.
- **Decrypt**$(CT, SK_{\mathbb{A}})$: This algorithm is run by the receiver. The inputs are a ciphertext CT which was encrypted under the set of attributes S, and the decryption key $SK_{\mathbb{A}}$ for access structure $\mathbb{A}$. The algorithm outputs the message M if $S \preceq \mathbb{A}$.

Next we give the security notion of KP-ABE. There are two security notions for KP-ABE, IND-CPA security and ANON-CPA security. The IND-CPA

security defines that a ciphertext does not reveal any information about the encrypted message, and the ANON-CPA defines that a ciphertext reveals nothing to the attribute set.

Definition 2.6. (IND-CPA Security)

We provide the IND-CPA security model for a KP-ABE scheme.

- Setup: A challenger runs the **Setup** algorithm to generate public parameters and master secret key, and gives the public parameters to the adversary.

- Phase 1: The adversary is allowed to issue polynomially many queries for private keys with access structures $\mathbb{A}_j$.

- Challenge: The adversary submits two equal-length messages M_0 and M_1 along with an attribute set S^*, where S^* should not satisfy any access structure $\mathbb{A}_j$ queried in Phase 1. The challenger flips a random coin b, and encrypts M_b with $\mathbb{A}^*$. The ciphertext is passed to the adversary.

- Phase 2: The adversary repeats the steps in Phase 1.

- Guess: The adversary outputs the guess $b' \in \{0, 1\}$ of b and wins the game if $b' = b$.

The advantage of the adversary in this game is defined as $Adv_{\mathcal{A}}^{IND-CPA} = |Pr[b' = b] - \frac{1}{2}|$. A KP-ABE scheme is said to be IND-CPA secure if for every polynomial-time adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{IND-CPA}$ is at most negligible.

Definition 2.7. (ANON-CPA Security)

We provide the ANON-CPA security models for a KP-ABE scheme.

- Setup: A challenger runs the **Setup** algorithm and gives the public parameters to the adversary.

- Phase 1: The adversary is allowed to issue polynomially many queries for private keys with access structures $\mathbb{A}_j$'s.

- Challenge: The adversary submits two equal-size sets S_0^* and S_1^* and a message M. The sets S_0^* and S_1^* should not satisfy any key that has been queried in Phase 1. The challenger flips a random coin b, and encrypts M with S_b^*. The ciphertext is passed to the adversary.

- Phase 2: The adversary repeats the steps in Phase 1.

- Guess: The adversary outputs the guess $b' \in \{0, 1\}$ of b and wins the game if $b' = b$.

The advantage of the adversary in this game is defined as $Adv_{\mathcal{A}}^{ANON-CPA} = |Pr[b' = b] - \frac{1}{2}|$. A KP-ABE scheme is said to be ANON-CPA secure if for every polynomial-time adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{ANON-CPA}$ is at most negligible.

The models can be easily extended to handle chosen-ciphertext attacks by allowing for decryption queries in Phase 1 and Phase 2.

2.7. Public Key Searchable Encryption from KP-ABE

In [22], Shen *et al.* give a generic construction of a public key searchable encryption scheme from a KP-ABE scheme supporting monotonic queries. The construction is an extension of that proposed in [6, 8], which builds a PEKS from a given identity-based encryption. Intuitively, the keywords can be regarded as attributes in KP-ABE. Additionally, we can regard the access structure as a search query associated with a trapdoor. Then, the KeyGen algorithm of KP-ABE can be performed as the Trapdoor algorithm of PEKS to generate a trapdoor for an access policy. The Encrypt algorithm of KP-ABE can be performed as the Encrypt algorithm of PEKS, the Decrypt algorithm of KP-ABE can be used for the Test algorithm of PEKS. More precisely, given a KP-ABE $(ABE.Setup, ABE.Encrypt, ABE.KeyGen, ABE.Decrypt)$, a PEKS is given as follows.

- Setup(1^λ): Run $ABE.Setup(1^\lambda) \rightarrow (PK, MSK)$, and output $(PK, SK) = (PK, MSK)$.

- Encrypt(PK, W): Choose a random message m, and compute $CT \leftarrow ABE.Encrypt(PK, W, m)$. Output $C = (CT, m)$.

- Trapdoor$(PK, SK, \mathbb{A})$: Generate a trapdoor $TK_{\mathbb{A}}$ for an access policy $\mathbb{A}$ as $TK_{\mathbb{A}} \leftarrow ABE.KeyGen(SK, \mathbb{A})$.

- Test$(PK, TK_{\mathbb{A}}, C = (CT, m))$: Output 1 if $m = ABE.Decrypt(CT, TK_{\mathbb{A}})$; output 0 otherwise.

The correctness of the generic construction is easily derived from the correctness of the underlying KP-ABE. On the other hand, like the generic construction given in [6, 8], the security of the PEKS relies upon the anonymity of the underlying KP-ABE, since the transformation regards keywords as the attributes in the underlying KPABE. Next we show that the construction is secure based on the security of the underlying KP-ABE.

Theorem 2.1. If the KP-ABE is ANON-CPA secure, then the PEKS form the KP-ABE is IND-CKA secure.

The detailed proof of this theorem can be refered to [22].

3. The Proposed Anonymous KP-ABE and PEKS

In this section, we first give a new anonymous KP-ABE scheme, and then give an efficient PEKS scheme via the transformation shown in Section 2.7.

3.1. The Proposed KP-ABE

Let $\mathbb{G}$ and $\hat{\mathbb{G}}$ be two multiplicative groups of prime order p, and $e : \mathbb{G} \times \hat{\mathbb{G}} \rightarrow \mathbb{G}_T$ be the bilinear map. The details of our scheme are described as follows.

3.1.1. Setup

Setup$(1^\lambda) \rightarrow (PK, MSK)$. Given a security parameter 1^λ. To generate the system parameters, the KGC performs the following steps:

1. Generate generators g and $\hat{g}$ of $\mathbb{G}$ and $\hat{\mathbb{G}}$, respectively.

2. Choose a hash functions $H : \{0,1\}^* \to \mathbb{Z}_p$.
3. Randomly select $\phi \in \mathbb{Z}_p$, and compute $h = g^\phi$ and $\hat{h} = \hat{g}^\phi$.
4. Randomly select $\alpha, \beta \in \mathbb{Z}_p$, and compute $U = e(g,\hat{g})^{\alpha(\beta-1)}$ and $V = e(g,\hat{g})^{\alpha\beta}$.
5. The public parameters is $PK = \{\mathbb{G}, \hat{\mathbb{G}}, e, g, U, V, h, H\}$, and the master secret key is $MSK = \{\hat{g}, \hat{g}^\alpha, \hat{h}\}$. Keep the master secret key MSK secret.

3.1.2. KeyGen

KeyGen$(PK, MSK, \mathbb{A} = (M, \rho, L)) \to SK_{\mathbb{A}}$. The KGC takes as input the master secret key and an LSSS access structure $\mathbb{A} = (M, \rho, L)$. Let M be an $\ell \times n$ matrix. The function ρ associates rows of M to attribute names. Let $L = (z_{\rho(1)}, ..., z_{\rho(\ell)})$ be the attribute values corresponding to $\rho(1), \ldots, \rho(\ell)$, respectively. Let τ be the set of distinct attribute names existing in the access structure matrix M. To generate a secret key associated with the access structure $\mathbb{A}$, the KGC performs the following steps:

1. Select a random vector $\vec{v} = (s, y_2, ..., y_n)$ where $s = 1$.
2. Compute $\lambda_i = M_i \vec{v}^\top$ for $i = 1$ to ℓ, where M_i is the i-th row of M.
3. Select random numbers $r_i \in \mathbb{Z}_p$ for $i = 1$ to ℓ.
4. Compute $\hat{\sigma}_i = \hat{g}^{H(\rho(i)\|z_{\rho(i)})} \cdot \hat{h}$ for $i = 1$ to ℓ.
5. For $i = 1$ to ℓ, compute $d_{i,0} = \hat{g}^{\alpha \cdot \lambda_i} \cdot \hat{\sigma_i}^{r_i}, \forall j \in \tau/\rho(i), Q_{i,j} = \hat{\sigma_j}^{r_i}$, and $d_{i,1} = \hat{g}^{r_i}$.
6. The secret key is $SK = (\{d_{i,0}, d_{i,1}, \{Q_{i,j}\}_{j \in \tau/\rho(i)}\}_{i=1}^{\ell})$.

3.1.3. Encrypt

Encrypt$(PK, S, m) \to CT$. The algorithm takes as input the public parameters PK, a message $m \in \mathbb{G}_T$, and an attribute value set $S = (v_1, ..., v_t)$, where v_i is the value of attribute name i. Let $\tilde{S}$ be the the set of attribute names from S. $\tilde{S}$ should be published with the ciphertext in order to decrypt. Note that the attribute values are not revealed to others. To generate a ciphertext of m associated with S, the algorithm performs the following steps:

1. Select a random number $k \in \mathbb{Z}_p$.

2. Compute $C_1 = V^k \cdot m = e(g, \hat{g})^{\alpha\beta k} \cdot m$.

3. Compute $C_2 = U^k = e(g, \hat{g})^{\alpha(\beta-1)k}$.

4. Compute $C_3 = g^k$.

5. Compute $\sigma_i = g^{H(i\|v_i)} \cdot h$ for $i = 1$ to t.

6. Compute $C_{4,i} = {\sigma_i}^k$ for $i = 1$ to t.

7. The ciphertext is $CT = (C_1, C_2, C_3, \{C_{4,i}\}_{i=1}^t, \tilde{S})$.

3.1.4. Decrypt

Decrypt$(CT, SK_{\mathbb{A}})$. If the set of attribute names $\tilde{S}$ does not satisfy the access structure $\mathbb{A}$, it outputs $\perp$. Otherwise, let $I \subseteq \{1, 2, ..., \ell\}$ be a set of indices and $\{\omega_i\}_{i\in I} \in \mathbb{Z}_p$ be a set of constants such that $\forall i \in I, \rho(i) \in \tilde{S}$ and $\prod_{i\in I} \omega_i M_i = (1, 0, ..., 0)$. Then we define $\Delta = \{x : \exists i \in I, \rho(i) = x\}$ and $\hat{f}(\Delta) = \prod_{x\in\Delta} \hat{\sigma_x}, f(\Delta) = \prod_{x\in\Delta} \sigma_x$. For each $i \in I$, compute $\hat{d_{i,0}} = d_{i,0} \cdot \prod_{x\in\Delta/\rho(i)} Q_{i,x} = \hat{g}^{\alpha\cdot\lambda_i} \hat{f}(\Delta)^{r_i}$. Compute $L = \prod_{x\in\Delta} C_{4,x} = \prod_{x\in\Delta} \sigma_x^k = f(\Delta)^k$. Then compute the following algorithm to decrypt the message m as follows:

$$
\begin{aligned}
Z &= \frac{e(C_3, \prod_{i\in I} \hat{d}_{i,0}^{\omega_i})}{e(L, \prod_{i\in I} d_{i,1}^{\omega_i})} \\
&= \frac{e(g^k, \hat{g}^{\alpha \sum_{i\in I} \lambda_i\omega_i} \hat{f}(\Delta)^{\sum_{i\in I} \omega_i r_i})}{e(f(\Delta)^k, \hat{g}^{\sum_{i\in I} r_i\omega_i})} \\
&= e(g, \hat{g})^{\alpha k} \\
m &= \frac{C_1}{C_2 \cdot Z} = \frac{e(g, \hat{g})^{\alpha\beta k} \cdot m}{e(g, \hat{g})^{\alpha(\beta-1)k} \cdot e(g, \hat{g})^{\alpha k}}
\end{aligned}
$$

The proposed KP-ABE scheme adopts the technique used in [34] to achieve fast decryption. The decryption in our KP-ABE needs only 2 parings, which is independent of the numbers of attributes in the ciphertext or the secret key. Furthermore, by adopting the transformation shown in Subsection 2.7, we obtain an efficient searchable encryption with constant pairings in the Test algorithm.

3.2. The PEKS from The Proposed KP-ABE

In Subsection 3.1, we proposed an anonymous KP-ABE with high efficiency. As mentioned in Subsection 2.7, an anonymous KP-ABE can be transformed

into a searchable encryption. In order to avoid the unnecessary repetition, we only give an intuition in this section. The main idea is to view keywords in PEKS as attributes in KP-ABE. First, we regard the access structure as a search query associated with a trapdoor. Then, the KeyGen algorithm of KP-ABE can be performed as the Trapdoor algorithm of PEKS to generate a trapdoor for an access policy. Since the underlying KP-ABE supports expressive access structures, and thus the proposed PEKS supports expressive queries. The Encrypt algorithm of KP-ABE can be slightly modified into the Encrypt algorithm of PEKS by encrypting a randomly chosen message M, and outputting M along with the ciphertext C outputted from the Encryption algorithm of KP-ABE. As for the Test algorithm, one uses the Decrypt algorithm of KP-ABE to decrypt C and obtain a message M', and then check whether $M = M'$. The reader is referred to Section 2.7 for details.

We give a simple example here. Assume that one wants to generate a ciphertext for keywords {School, Dep , Deg} = {NSYSU, CSE, Master}, where {School, Dep, Deg} are the keyword names, and NSYSU, CSE, Master are the keyword values for School, Dep, Deg, respectively. Then the corresponding ciphertext will be

$$CT = \begin{pmatrix} M, C_1 = V^k \cdot M, & C_2 & = U^k, & C_3 = g^k \\ C_{4,School} = \sigma_{School}^k & C_{4,Dep} = \sigma_{Dep}^k & C_{4,Deg} = \sigma_{Dep}^k & \end{pmatrix},$$

where $\sigma_{School} = g^{H(School \| NSYSU)}h, \sigma_{Dep} = g^{H(Dep \| CSE)}h, \sigma_{Dep} = g^{H(Deg \| Master)}h$. We can observe that, all the components except M is the ciphertext elements generated from the Encryption algorithm shown in Section 3.1.3. As for generating trapdoors, assume that we want to generate a search token with search pattern "Dep = CSE $\wedge$ Deg = Master". Then we need to represent the search pattern with an LSSS access structure (M, ρ, L), where

$$M = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix}, \rho(1) = Dep, \rho(2) = Deg, L = (CSE, Master).$$

Note that M may not be unique, any M satisfying the requirements defined

in Definition 2.3 is valid. Next, we can compute λ_1, λ_2 as described in Section 3.1.2, and compute the search token

$$TK = \begin{pmatrix} \{d_{1,0} = \hat{g}^{\alpha\lambda_1}\hat{\sigma}_1^{r_1} & d_{1,1} = \hat{g}^{r_1} & \{Q_{1,2} = \hat{\sigma}_2^{r_1}\}\} \\ \{d_{2,0} = \hat{g}^{\alpha\lambda_2}\hat{\sigma}_2^{r_2} & d_{2,1} = \hat{g}^{r_2} & \{Q_{2,1} = \hat{\sigma}_1^{r_2}\}\} \end{pmatrix},$$

where $\hat{\sigma}_1 = \hat{g}^{H(Dep\|CSE)}\hat{h}, \hat{\sigma}_2 = \hat{g}^{H(Deg\|Master)}\hat{h}$. We can see that, TK is of the same form of the private key of the proposed KP-ABE. Therefore, one can use TK as a private key to perform the Decryption algorithm at Section 3.1.4, and check whether the decrypted result equals to M, and this is how the Test algorithm work.

4. Security Proofs

In this section, we will prove the indistinguishability of encryption under chosen-plaintext attacks (IND-CPA security) and the anonymity of encryption under chosen-plaintext attacks (ANON-CPA security) of our KP-ABE scheme.

4.1. IND-CPA Security

In this subsection, we will prove the IND-CPA security for the proposed KP-ABE scheme.

Theorem 4.1. The proposed KP-ABE scheme is IND-CPA secure if the DBDH-3 problem is hard.

Proof. Assume that there is an adversary $\mathcal{A}$ who has the advantage ϵ in winning the IND-CPA game. We will construct an algorithm $\mathcal{C}$ that can solve the DBDH-3 problem. Taking as input $(g, \hat{g}, g^a, g^b, g^c, \hat{g}^a, \hat{g}^b, Y)$, where Y is either $e(g, \hat{g})^{abc}$ or a random element in $\mathbb{G}_T$, $\mathcal{C}$ performs as follows:

Setup. $\mathcal{C}$ simulates the phase as follows.

1. Choose a hash function $H : \{0,1\}^* \to \mathbb{Z}_p$.
2. Randomly select $\phi \in \mathbb{Z}_p$, and compute $h = g^\phi$ and $\hat{h} = \hat{g}^\phi$.
3. Compute $U = e(g^b/g, \hat{g}^a)$ and $V = e(g^b, \hat{g}^a)$.

4. Publish the public parameters $PK = \{\mathbb{G}, \hat{\mathbb{G}}, e, g, U, V, h, H\}$, and keep secret the master secret key $MSK = \{\hat{g}, \hat{g}^a, \hat{h}\}$.

Phase 1. In this phase, $\mathcal{A}$ can query for private keys with access structures $\mathbb{A}$. Since $\mathcal{C}$ has the master secret key MSK, $\mathcal{C}$ is able to perform $KeyGen$ as the proposed scheme to answer the queries. I.e., we can follow the **KeyGen** algorithm shown in Section 3.1.2 to generate trapdoors.

Challenge. On inputting two equal-length messages $m_0^*, m_1^* \in \mathbb{G}_T$ and a target attribute set $S^* = (v_1, ..., v_t)$ which does not satisfy the access structures queried in phase 1, $\mathcal{C}$ performs as follows:

1. Choose $\beta \in \{0, 1\}$.
2. Set $C_3^* = g^c$.
3. For $i = 1$ to t, compute $C_{4,i}^* = (g^c)^{H(i\|v_i)+\phi}$.
4. Compute $C_1^* = Y \cdot m_\beta$ and $C_2^* = \frac{Y}{e(g^c, \hat{g}^a)}$.
5. Send $C^* = (C_1^*, C_2^*, C_3^*, \{C_{4,i}^*\}_{i=1}^t)$.

Phase 2. In this phase, $\mathcal{A}$ can query for private keys with access structures $\mathbb{A}$ which cannot be satisfied by the target attribute set S^*. Since $\mathcal{C}$ has the master secret key MSK, $\mathcal{C}$ is able to perform $KeyGen$ as the proposed scheme to answer the queries.

Challenge. $\mathcal{A}$ outputs the guess $\beta' \in \{0, 1\}$ and wins the game if $\beta' = \beta$. $\mathcal{C}$ outputs 1 if $\mathcal{A}$ wins the game.

If $Y = e(g, \hat{g})^{abc}$, then $C_1^* = e(g, \hat{g})^{abc} \cdot m_\beta = V^c \cdot m_\beta, C_2^* = e(g, \hat{g})^{a(b-1)c} = U^c, C_3^* = g^c, C_{4,i}^* = (g^c)^{H(i\|v_i)+\phi} = \left(g^{H(i\|v_i)} \cdot h\right)^c$. Thus, C^* is well-formed. If $Y \in_R \mathbb{G}_T$, C^* is not well-formed, and thus the advantage ϵ of $\mathcal{A}$ is negligible. Besides, the challenger is able to answer $KeyGen$ queries with any access structure $\mathbb{A}_j$ because the challenger has the master secret key MSK. The adversary

is also allowed to query the **Challenge** phase with any attribute set. Therefore, if $\mathcal{A}$ has non-negligible advantage ϵ in winning the game, $\mathcal{C}$ is able to solve the DBDH-3 problem in the same advantage. That is

$$\begin{aligned} & \Big|\Pr[\mathcal{C}(g,\hat{g},g^a,g^b,g^c,\hat{g}^a,\hat{g}^b,e(g,\hat{g})^{abc})=1] \\ & -\Pr[\mathcal{C}(g,\hat{g},g^a,g^b,g^c,\hat{g}^a,\hat{g}^b,Y\overset{\$}{\leftarrow}\mathbb{G}_T)=1]\Big| \\ = & \Big|\Pr[b'=b|Y=e(g,\hat{g})^{abc}]-\Pr[b'=b|e(g,\hat{g})]^{abc}|Y\overset{\$}{\leftarrow}\mathbb{G}_T]\Big| \\ \geq & \epsilon. \end{aligned}$$

□

4.2. ANON-CPA Security

In this subsection, we will prove the ANON-CPA security for the proposed KP-ABE scheme.

Theorem 4.2. The proposed KP-ABE scheme is ANON-CPA secure if the DBDH-3 problem is hard.

Proof. Assume that there is an adversary $\mathcal{A}$ who has the advantage ϵ in winning the ANON-CPA game. We will construct an algorithm $\mathcal{C}$ that can solve the DBDH-3 problem. Taking as input $(g,\hat{g},g^a,g^b,g^c,\hat{g}^a,\hat{g}^b,Y)$, where Y is either $e(g,\hat{g})^{abc}$ or a random element in $\mathbb{G}_T$, $\mathcal{C}$ performs as follows:

Setup. $\mathcal{C}$ simulates the phase as follows.

1. Choose a hash function $H:\{0,1\}^*\rightarrow\mathbb{Z}_p$.
2. Randomly select $\phi\in\mathbb{Z}_p$, and compute $h=g^\phi$ and $\hat{h}=\hat{g}^\phi$.
3. Compute $U=e(g^b/g,\hat{g}^a)$ and $V=e(g^b,\hat{g}^a)$.
4. Publish the public parameters $PK=\{\mathbb{G},\hat{\mathbb{G}},e,g,U,V,h,H\}$, and keep secret the master secret key $MSK=\{\hat{g},\hat{g}^a,\hat{h}\}$.

Phase 1. In this phase, $\mathcal{A}$ can query for private keys with access structures $\mathbb{A}$. Since $\mathcal{C}$ has the master secret key MSK, $\mathcal{C}$ is able to perform $KeyGen$ as the proposed scheme to answer the queries. I.e., we can follow the **KeyGen**

algorithm shown in Section 3.1.2 to generate trapdoors.

Challenge. On inputting a message $m \in \mathbb{G}_T$ and two attribute sets of equal size $S_0^* = (v_1^{(0)}, ..., v_t^{(0)}), S_1^* = (v_1^{(1)}, ..., v_t^{(1)})$ where $(S_0^* \preceq \mathbb{A} \wedge S_1^* \preceq \mathbb{A})$ or $(S_0^* \npreceq \mathbb{A} \wedge S_1^* \npreceq \mathbb{A})$ for every $\mathbb{A}$ queried in phase 1, $\mathcal{C}$ performs as follows:

1. Choose $\beta \in \{0, 1\}$.
2. Set $C_3^* = g^c$.
3. For $i = 1$ to t, compute $C_{4,i}^* = (g^c)^{H(i||v_i^{(\beta)})+\phi}$.
4. Compute $C_1^* = Y \cdot m$ and $C_2^* = \frac{Y}{e(g^c, \hat{g}^a)}$.
5. Send $C^* = (C_1^*, C_2^*, C_3^*, \{C_{4,i}^*\}_{i=1}^t)$.

Phase 2. In this phase, $\mathcal{A}$ can query for private keys with access structures $\mathbb{A}$ where $(S_0^* \preceq \mathbb{A} \wedge S_1^* \preceq \mathbb{A})$ or $(S_0^* \npreceq \mathbb{A} \wedge S_1^* \npreceq \mathbb{A})$. Since $\mathcal{C}$ has the master secret key MSK, $\mathcal{C}$ is able to perform $KeyGen$ as the proposed scheme to answer the queries.

Guess. $\mathcal{A}$ outputs the guess $\beta' \in \{0, 1\}$ and wins the game if $\beta' = \beta$. $\mathcal{C}$ outputs 1 if $\mathcal{A}$ wins the game.

If $Y = e(g, \hat{g})^{abc}$, then $C_1^* = e(g, \hat{g})^{abc} \cdot m = V^c \cdot m, C_2^* = e(g, \hat{g})^{a(b-1)c} = U^c, C_3^* = g^c, C_{4,i}^* = (g^c)^{H(i||v_i^{(\beta)})+\phi} = \left(g^{H(i||v_i^{(\beta)})} \cdot h\right)^c$. Thus, C^* is well-formed. If $Y \in_R \mathbb{G}_T$, C^* is not well-formed, and thus the advantage ϵ of $\mathcal{A}$ is negligible. Besides, the challenger is able to answer $KeyGen$ queries with any access structure $\mathbb{A}_j$ because the challenger has the master secret key MSK. The adversary is also allowed to query the **Challenge** phase with any attribute set. Therefore, if $\mathcal{A}$ has non-negligible advantage ϵ in winning the game, $\mathcal{C}$ is able to solve the

DBDH-3 problem in the same advantage. That is

$$\begin{aligned} & \Big|\Pr[\mathcal{C}(g,\hat{g},g^a,g^b,g^c,\hat{g}^a,\hat{g}^b,e(g,\hat{g})^{abc})=1] \\ & -\Pr[\mathcal{C}(g,\hat{g},g^a,g^b,g^c,\hat{g}^a,\hat{g}^b,Y\overset{\$}{\leftarrow}\mathbb{G}_T)=1]\Big| \\ = & \Big|\Pr[b'=b|Y=e(g,\hat{g})^{abc}]-\Pr[b'=b|e(g,\hat{g})]^{abc}|Y\overset{\$}{\leftarrow}\mathbb{G}_T]\Big| \\ \geq & \epsilon. \end{aligned}$$

□

4.3. The IND-CKA Security of the Proposed PEKS

In this subsection, we will prove the IND-CKA security for the proposed PEKS scheme.

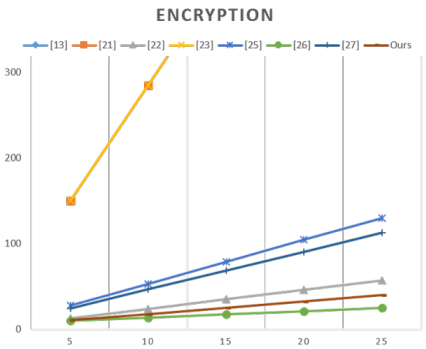
Theorem 4.3. The SE from the proposed KP-ABE is IND-CKA secure.

Proof. Based on Theorem 2.1, we have that, if the underlying KP-ABE is ANON-CPA secure, then the PEKS from the KP-ABE is semantically secure. From Theorem 4.2, we have that the proposed KP-ABE is ANON-CPA secure. As a result, the SE from the proposed KP-ABE is IND-CKA secure. □

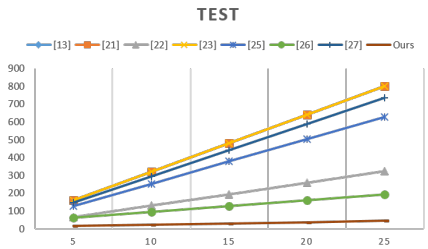
5. Comparisons

In this section, we compare our scheme with existing related schemes which support monotonic queries [25, 21, 13, 23, 26, 27]. TABLE 1 and TABLE 2 present the comparisons of the properties and asymptotic analysis on performance, respectively. In TABLE 1, we compare the following properties:

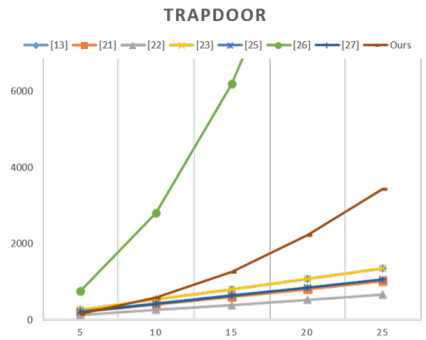
- Group Order: The schemes in [25, 26, 27] are based on prime order bilinear groups, while the schemes in [21, 13, 23] are based on composite order bilinear groups which suffer from the heavy computation cost. According to [14], 3072 bits are required to store an element in a composite-order group. However, it requires 256 bits in prime-order groups which is 12 times less than that in composite-order groups.



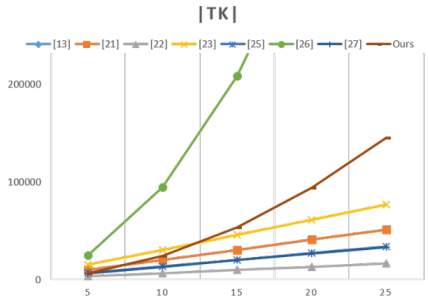
(a) Encryption



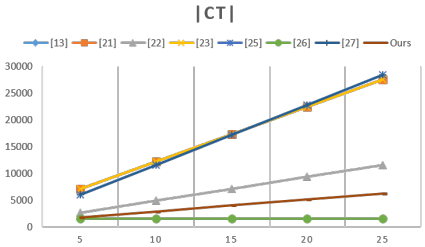
(b) Test



(c) Trapdoor



(d) $|TK|$



(e) $|CT|$

Figure 2: Bar Charts for Table 2

Table 1: Property Comparison

	Group Order	Feasibility in Test	Correctness in Test	Selective/ Full Security	STD/ ROM	Keyword Space	Without Online TTP
[13]	Composite	No	Yes	Full	STD	Pol	Yes
[21]	Composite	No	Yes	Full	STD	Pol	Yes
[22]	Prime	No	Yes	Selective	STD	Exp	Yes
[23]	Composite	No	Yes	Full	STD	Pol	Yes
[25]	Prime	Yes	Yes	Selective	STD	Exp	No
[26]	Prime	Yes	No	Selective	STD	Exp	No
[27]	Prime	Yes	Yes	Selective	STD	Exp	Yes
Ours	Prime	Yes	Yes	Full	STD	Exp	Yes

- Feasibility of Test: [13, 21, 22, 23] are public key searchable encryptions based on anonymous KP-ABE. However, the Test algorithms of these schemes need to correlate the elements of key and the elements of ciphertext by attributes. The problem of [13, 21, 22, 23] is that the Test algorithms will not be conducted successfully because of the exponential complexity in finding the correlation between the corresponding key and ciphertext, as mentioned in Introduction.

- Correctness of Test: To achieve constant-size ciphertext, the Test algorithm in [26] needs to aggregate the ciphertext components for all attributes into a group element. However, in this way, the Test algorithm requires that all attributes in a ciphertext should appear in the access structure, or it would fail. For instance, a ciphertext associated with keyword names $\{A, B, C\}$ will not be searched by a trapdoor associated with an accesss structure $\{A \wedge (B \vee D)\}$, because the keyword name C does not appear in the access structure.

- Selective/Full Security: In a selective security model, the adversary is asked to give the target before Setup phase, while in a full security model,

Table 2: Asymptotic Performance Comparison

	Encryption	Test	Trapdoor	$\|TK\|$	$\|CT\|$
[13]	$(2t+2)T_a$ $+(2t+1)T_s+T_G$	$(2\|I\|-1)T_a$ $+\|I\|T_G+(2\|I\|)T_p$	$4\ell T_a$ $+4\ell T_s$	$2\ell\|\mathbb{G}\|$	$(t+1)\|\mathbb{G}\|$ $+1\|\mathbb{G}_T\|$
[21]	$(2t+2)T_a$ $+(2t+1)T_s+T_G$	$2\|I\|T_a$ $\|I\|T_G+(2\|I\|)T_p$	$4\ell T_a$ $+3\ell T_s$	$2\ell\|\mathbb{G}\|$	$(t+1)\|\mathbb{G}\|$ $+1\|\mathbb{G}_T\|$
[22]	$2tT_a$ $+(3t+1)T_s+T_G$	$\|I\|T_G$ $+(3\|I\|)T_p$	$2\ell T_a$ $+5\ell T_s$	$3\ell\|\mathbb{G}\|$	$(2t+1)\|\mathbb{G}\|$ $+1\|\mathbb{G}_T\|$
[23]	$(2t+1)T_a$ $+(2t+1)T_s+T_G$	$(2\|I\|-1)T_a$ $+\|I\|T_G+(2\|I\|)T_p$	$4\ell T_a$ $+4\ell T_s$	$3\ell\|\mathbb{G}\|$	$(t+1)\|\mathbb{G}\|$ $+1\|\mathbb{G}_T\|$
[25]	$2tT_a$ $+(7t+1)T_s$ $+T_G$	$(7\|I\|-1)T_a$ $+(\|I\|+1)T_G$ $+(6\|I\|+1)T_p$	$3\ell T_a$ $+(8\ell+1)T_s$ $+T_p$	$(6\ell+2)\|\mathbb{G}\|$	$(5t+1)\|\mathbb{G}\|$ $+1\|\mathbb{G}_T\|$
[26]	$(t+1)T_a$ $+(t+7)T_s$ $+T_G$	$(4\ell+6\|I\|-12)T_a$ $+(4\ell+5\|I\|-4)T_s$ $+2T_G+7T_p$	$3\ell T_a$ $+(5\ell^2+3\ell+1)T_s$ $+T_p$	$(4\ell^2+2\ell$ $+2)\|\mathbb{G}\|$	$6\|\mathbb{G}\|+1\|\mathbb{G}_T\|$
[27]	$3T_a$ $+(6t+2)T_s+T_G$	$5\|I\|T_a+(5\|I\|+1)T_G$ $+6\|I\|T_p$	$2\ell T_a$ $+8\ell T_s$	$6\ell\|\mathbb{G}\|$	$(5t+1)\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$
Ours	tT_a $+(2t+1)T_s+2T_G$	$(\ell+\|I\|)T_a$ $+(2\|I\|)T_s+2T_p$	$2\ell T_a$ $+(\ell^2+\ell)T_s$	$(\ell^2+\ell)\|\mathbb{G}\|$	$(t+1)\|\mathbb{G}\|$ $+2\|\mathbb{G}_T\|$

the adversary is allowed to give the target in Challenge phase. Obviously, full security is stronger than selective security.

- Standard/Random Oracle Model: The security proven in the standard model is stronger than that proven in the random oracle model, since the random oracle model is widely believed to be a heuristic model. We use STD to denote "standard model" and ROM to denote "random oracle model".

- Keyword Space: We use "Exp" to denote exponentially large keyword

space and "Pol" to denote polynomially large keyword space. "Exp" is better than "Pol" because we can dynamically add any keyword in the system.

- Without Online TTP: In [25, 26], an online and trusted third party is required to generate trapdoors for users. It is an impractical and unreasonable assumption in cryptography.

The existing public key searchable encryption schemes with monotonic queries have some drawbacks as presented in TABLE 1. Our scheme is the first one that can overcome these drawbacks. In TABLE 2, the number of attributes which are used in Test algorithm, attributes which are associated with trapdoor, attributes which are associated with ciphertext are denoted as $|I|$, ℓ, and t, respectively. Besides, the cost of a group operation, the cost of a scalar multiplication in $\mathbb{G}(\hat{\mathbb{G}})$, the cost of a scalar multiplication in $\mathbb{G}_T$, and the cost of a pairing operation are denoted as T_a, T_s, T_G, and T_p, respectively. We can observe that our scheme only requires 2 pairings in the Test algorithm which is independent of the number of attributes used in ciphertexts and trapdoors. To the best of our knowledge, it is the most efficient public-key encryption with keyword search supporting monotonic query in the literature. The corresponding bar charts for Table 2 are shown in Figure 2a to 2e. Figure 2a, 2b, 2c show the computation cost for Encrypt, Test, Trapdoor algorithm, respectively. Figure 2d and 2e demonstrate the size of CT and TK, respectively. One can see that in Figure 2b, our scheme is the most lightweight one, since only two pairings are needed. However, the efficiency of the Trapdoor algorithm and the size of TK are worse than others, since their complexity are in the square of the keywords used in the algorithms. Meanwhile, one may also observe that in Figure 2a, 2b, 2e, the performance of [13, 21, 23] are worse than others, due to the usage of composite order groups. As mentioned in [14], the computation cost of pairings in composite order groups is much more slower than those in prime order groups. As a result, the schemes of [13, 21, 23] might not be suitable for practical usage.

For evaluate the real-world performance, we implement our scheme and the schemes[2] in [22, 25, 26, 27] for the comparison on the storage and computation overhead. We implement these schemes with MNT curves [35] with $|q|$ = 160 bits. The reason for comparing our work with [22, 25, 26] is that [22, 25, 26] are PEKS supporting expressive search query built under prime-order groups, while other existing works are built under composite-order groups. As stated in [14], the performance of composite-order groups is notoriously worse than that of prime-order groups in both computation and storage overhead. The environment of the implementation is shown in TABLE 3 and the implementation result is shown in TABLE 4. Besides, the corresponding bar charts are shown in Figure 4a to 4d. Since the real-world performance relates to the keywords of ciphertexts and the predicates for trapdoor, we assume a scenario for the implementation as follows: A user wants to search for a file whose keywords satisfies the formula {"$School$ = $NSYSU$" $\wedge$ (("$Department$ = CSE" $\wedge$ "$Degree$ = $Masters$") $\vee$ "$Position$ = $Teacher$")}, as shown in Figure 3[3]. Besides, there is an encrypted file stored in the cloud which is attached with the keywords $\{School, Position\}$ = $\{NSYSU, Teacher\}$. The scheme would check whether this encrypted file is satisfied to the user's query or not. The encryption time of our scheme is only 50% of that of [25], 77% of that of [26], and 61% of that of [22]. For the computation overhead on Test algorithm[4], ours reduces 86% , 76%, 74% of that of [25], [26], [22], respectively. As for the ciphertext size, the performance of [26] is slightly better than ours since their ciphertext length is independent of the number of keywords. However, their scheme suffers

[2] All codes are available via
https://github.com/yftseng/Implementation-of-PEKS

[3] To encode a policy into an LSSS matrix, we apply the algorithm shown in Appendix G of [36].

[4] Note that the performing time of Test is related to the number of attributes used in Test algorithm and that associated with a trapdoor. Therefore, the time would change according to the scenario.

from some additional restriction, which may make their scheme inflexible and impractical. The reader is referred to Remark 3 for details. Compared with [25], the ciphertext length of our scheme is 40% shorter than that of [25]. Besides, as shown in TABLE 2, the asymptotic complexity of the ciphertext of our scheme and other composite-order-based schemes are the same. However, for security consideration, the length of an element in composite-order groups is much longer than that in prime-order groups. Therefore, the ciphertext length of our scheme shorter than those of [21, 13, 23]. Though the trapdoor length of our scheme is shorter than others in TABLE 4, the size would be grower faster than others since our trapdoor length is $O(\ell^2)$.

Besides, we also compared our work with [27] proposed by Hao et al. recently, which is the most expressive and efficient anonymous KP-ABE to the best of our knowledge. Actually, anonymous KP-ABE [27, 37, 38] drew much less attention then its dual variant, i.e. ciphertext-policy attribute-based encryption with hidden policy [39, 40, 41, 42, 43, 44]. By applying Hao et al.s anonymous KP-ABE scheme into the transformation shown in [22], we then obtain a PEKS scheme, whose comparison result is shown in TABLE 4 as well. For encryption time, ours is only 49% of Hao et al.'s under MNT curves. For the cost of Test algorithm, our scheme reduces 83% cost compared to Hao et al.'s scheme. For the storage overhead, the ciphertext/ trapdoor length are 63%/83% of those in Hao et al.'s scheme under MNT curves.

Remark 2. Note that [21, 13, 23] have the infeasibility problem in the Test algorithm as mentioned above. In TABLE 2, we suppose that the correlation between the ciphertext and the secret key has been successfully accomplished. We only consider the cost of the computations in their Test algorithms.

Remark 3. Note that the ciphertext length in [26] is independent of the number of keywords (t) in the ciphertext. However, in their scheme, the Test algorithm requires all the attributes in a ciphertext to appear in the access structure on the secret key. That is, all the attributes in the ciphertext must be used to decrypt

the ciphertext. This restriction will make the Test algorithm fail under certain circumstances. For instance, if we make the query as Fig. 3 to search a ciphertext with keywords $\{School, Position, Gender\} = \{NSYSU, Teacher, Female\}$, then Test algorithm would fail even though the keywords match the query. This additional restriction makes the scheme of [26] inflexible and impractical.

Remark 4. In [45], the authors proved the lower bound of ciphertexts for anonymous broadcast encryption. Their theorem states that the ciphertext must be linear to the size of the receiver set, since the security definition for anonymity requires that no information about the receiver set should be leaked from a ciphertext. A similar statement has also been shown in [46]. Therefore, we believe that, in an anonymous KP-ABE scheme supporting monotonic access structure, the ciphertext length of a ciphertext would be linear to the number of the attributes corresponding to the ciphertext; otherwise some problem would occur in either security or correctness. The reason is that the predicate supported by KP-ABE seems much more complicated then that of broadcast encryption. Note that, though [26] achieves constant ciphertext size, their scheme fails in Test algorithm for certain cases (as stated in Remark 3), and hence it may not achieve "correctness".

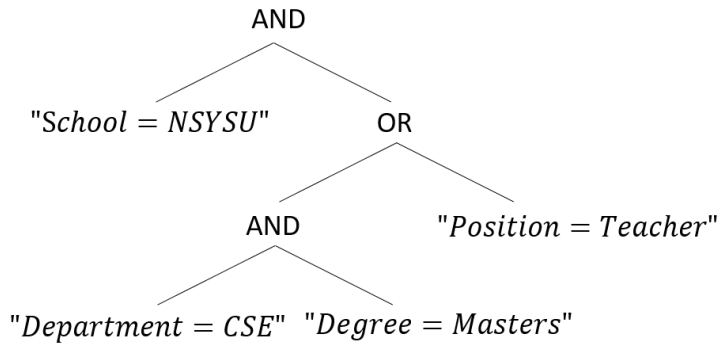
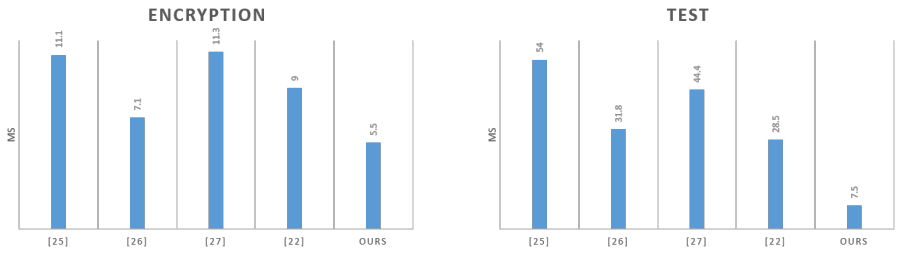
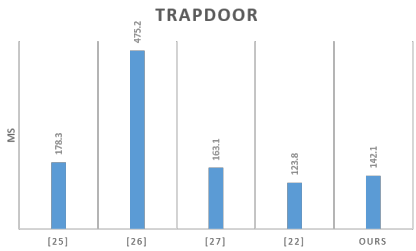


Figure 3: Example for Query

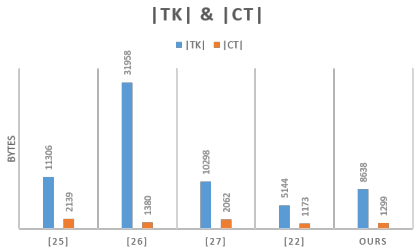


(a) Encryption

(b) Test



(c) Trapdoor



(d) $|TK|$ & $|CT|$

Figure 4: The Time Cost for Encryption/Test/Trapdoor and Size for TK/CT

Table 3: The environment of the implementation

	Specification
OS	Ubuntu 18.04 LTS
CPU	i7-4790 3.6GHz
RAM	8 Gb
Language	Python 3.6
Library	Charm-Crypto v0.50 [47]

Table 4: Implementation Results: MNT Curves

	Encryption	Test	Trapdoor	$\|TK\|$	$\|CT\|$
[25]	11.1 ms	54 ms	214.1ms	11306 Bytes	2139 Bytes
[26]	7.1 ms	31.8 ms	509.9ms	31958 Bytes	1380 Bytes
[27]	11.3 ms	44.4 ms	199.2ms	10298 Bytes	2062 Bytes
[22]	9 ms	28.5 ms	123.8ms	5144 Bytes	1173 Bytes
Ours	5.5 ms	7.5 ms	174.3ms	8638 Bytes	1299 Bytes

6. Conclusion

Public key searchable encryption with expressive queries is necessary for people to search encrypted files, due to the widely usage of cloud services nowadays. However, the existing schemes which support monotonic queries suffer from problems such as heavy computation cost, infeasibility or incorrect in testing, weaker security notion, polynomial keyword space, and the requirement of online trusted third party. In this work, we focus on constructing a new public key searchable encryption to overcome the aforementioned drawbacks. We first proposed an expressive anonymous KP-ABE with fast decryption with provable security in the standard model. By applying the Shen *et al.*'s transformation to our anonymous KP-ABE, we obtain an efficient PEKS scheme supporting monotonic search queries. The pairings needed in the testing procedure of our scheme is independent of the number of keywords and the size of the search

query; only two pairings are required. To further evaluate the performance, we implement our scheme and other with Python under MNT curves. As shown in TABLE 4, the computation cost is reduced around 74%-86% compared with other works. Besides, the ciphertext of our scheme is shorter then most of other works. To the best of our knowledge, the proposed scheme is the most efficient PEKS scheme supporting monotonic query. In addition, we believe that our proposed anonymous KP-ABE is of independent interest due to its anonymity and efficiency in decryption. For the future work, one could consider the forward and backward secrecy for PEKS, which are both important properties in cloud environment.

Acknowledgments

This work was partially supported by Taiwan Information Security Center at National Sun Yat-sen University (TWISC@NSYSU) and the Ministry of Science and Technology of Taiwan under grants MOST 110-2218-E-110-007-MBK and 110-2221-E-004 -003 - . It also was financially supported by the Information Security Research Center at National Sun Yat-sen University in Taiwan and the Intelligent Electronic Commerce Research Center from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

Appendix A. Anonymous and Non-Anonymous KP-ABE

By "anonymous KP-ABE", we means that, a ciphertext reveals no information about the attribute set used for encryption in a KP-ABE scheme. We have proven that our KP-ABE scheme shown in Section 3.1 achieves anonymity in Section 4.2. We further show a example for KP-ABE that is not anonymous. Here we take the scheme in [34] as an example. We briefly introduce the algorithms of [34].

Setup(1^λ). Assume there are n attributes in the system. For simplicity, we may assume that the universe $\mathcal{U} = \{1, \ldots, N\}$. Taking as input a security parameter, the algorithm performs as follows.

1. Generate a description $(e, \mathbb{G}, \mathbb{G}_T, p, g)$ for a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, where $|\mathbb{G}| = |\mathbb{G}_T|$ is a prime p and g is a generator of $\mathbb{G}$.
2. For each attribute i, randomly choose h_i from $\mathbb{G}$.
3. Choose α randomly from $\mathbb{Z}_p$, and compute $Y = e(g, g)^\alpha$.
4. Output the public parameter $PK = \{e, \mathbb{G}, \mathbb{G}_T, p, g, Y, h_1, \ldots, h_N\}$ and the master secret key $MSK = \alpha$.

Encrypt(PK, S, m). Taking as inputs the public parameter PK, an attribute set S, and a message m, the algorithm performs as follows.

1. Randomly choose s from $\mathbb{Z}_p$.
2. Compute $C = m \cdot Y^s$ and $C' = g^s$.
3. For $x \in S$, compute $C_x = h_x^s$.
4. Output $CT_S = (C, C', \{C_x\}_{x \in S})$.

KeyGen$(MSK, \mathbb{A})$. Taking as inputs the master secret key $MSK = \alpha$ and an access structure $\mathbb{A} = (M, \rho)$, where $M \in \mathbb{Z}_p^{\ell \times n}$, $\rho : [1, \ell] \to \mathcal{U}$ is a map from the index of the row in M to the attribute universe, the algorithm performs as follows. Let Γ be the distinct attributes used in $\mathbb{A}$, i.e., $\Gamma = \{y \mid \exists i \in [1, \ell], \rho(i) = y\}$.

1. Set $\vec{v} = (\alpha, v_2, \ldots, v_n)$ for randomly chosen $v_2, \ldots, v_n \in \mathbb{Z}_p$.
2. For $i = 1, dots, \ell$, compute $\lambda_i = M \cdot \vec{v}^\top$.
3. Randomly choose $r_1, \ldots, r_\ell \in \mathbb{Z}_p$.
4. For $i = 1, dots, \ell$, compute

$$D_i = g^{\lambda_i} \cdot h_{\rho(i)}^{r_i}, R_i = g^{r_i}, \forall y \in \Gamma / \rho(i), Q_{i,y} = h_y^{r_i}.$$

5. Output the private key $SK_\mathbb{A} = \{D_i, R_i, \{Q_{i,y}\}_{y \in \Gamma / \rho(i)}\}_{i=1}^{\ell}$.

Decrypt$(CT_S, SK_\mathbb{A})$. Taking as inputs a ciphertext CT_S for an attribute set S and a private key $SK_\mathbb{A}$ for an access structure $\mathbb{A} = (M \in \mathbb{Z}_p^{\ell \times n}, \rho : [1, \ell] \to \mathcal{U})$ the algorithm performs as follows.

1. Find sets $I \subseteq [1, \ell]$ and $\{\omega_i\}_{i \in I}$ such that for all $i \in I, \rho(i) \in S$, and $\sum_{i \in I} \omega_i \cdot M_i = (1, 0, \ldots, 0)$, where M_i denotes the i-th row of M.
2. Define $\delta = \{x \mid \exists i \in I, \rho(i) = x\}$.
3. For $i \in I$, compute $D_i' = D_i \cdot \prod_{x \in \delta / \rho(i)} Q_{i,x}$.
4. Compute $L = \prod_{x \in \delta} C_x$.
5. Compute $Y_S = e(g,g)^{\alpha s} = e(C', \prod_{i \in I} (D_i')^{\omega_i}) / e(\prod_{i \in I} R_i^{\omega_i}, L)$, and recover m from C.

It is easy to see that $(g, h_x, C' = g^s, C_x = h_x^s)$ is a DDH-tuple. Therefore, given a ciphertext $CT_S = (C, \{C_x\}_{x \in S})$, one can test whether a attribute z is in S by checking

$$e(g, C_x) \stackrel{?}{=} e(h_z, C'), \quad \text{for } x \in S.$$

References

[1] J. Aikat, A. Akella, J. S. Chase, A. Juels, M. K. Reiter, T. Ristenpart, V. Sekar, M. Swift, Rethinking security in the era of cloud computing, IEEE Security Privacy 15 (3) (2017) 60–69. doi:10.1109/MSP.2017.80.

[2] N. Chen, J. Li, Y. Zhang, Y. Guo, Efficient cp-abe scheme with shared decryption in cloud storage, IEEE Transactions on Computers 71 (1) (2022) 175–184. doi:10.1109/TC.2020.3043950.

[3] J. Li, Y. Zhang, J. Ning, X. Huang, G. S. Poh, D. Wang, Attribute based encryption with privacy protection and accountability for cloudiot, IEEE Transactions on Cloud Computing (2020) 1–1doi:10.1109/TCC.2020.2975184.

[4] J. Li, X. Lin, Y. Zhang, J. Han, Ksf-oabe: Outsourced attribute-based encryption with keyword search function for cloud storage, IEEE Transactions on Services Computing 10 (5) (2017) 715–725. doi:10.1109/TSC.2016.2542813.

[5] D. X. Song, D. Wagner, A. Perrig, Practical techniques for searches on encrypted data, in: Proceeding 2000 IEEE Symposium on Security and Privacy. S P 2000, 2000, pp. 44–55.

[6] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, in: Advances in Cryptology - EUROCRYPT 2004, 2004, pp. 506–522.

[7] R. Curtmola, J. Garay, S. Kamara, R. Ostrovsky, Searchable symmetric encryption: Improved definitions and efficient constructions, in: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS '06, ACM, New York, NY, USA, 2006, pp. 79–88. doi:10.1145/1180405.1180417.
URL http://doi.acm.org/10.1145/1180405.1180417

[8] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, H. Shi, Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions, in: Advances in Cryptology – CRYPTO 2005, 2005, pp. 205–222.

[9] D. Boneh, B. Waters, Conjunctive, subset, and range queries on encrypted data, in: Theory of Cryptography, Springer Berlin Heidelberg, 2007, pp. 535–554.

[10] J. Katz, A. Sahai, B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, in: Advances in Cryptology – EUROCRYPT 2008, 2008, pp. 146–162.

[11] C.-I. Fan, V. S.-M. Huang, H.-M. Ruan, Arbitrary-state attribute-based encryption with dynamic membership, IEEE Transactions on Computers 63 (8) (2014) 1951–1961. doi:10.1109/TC.2013.83.

[12] S.-Y. Huang, C.-I. Fan, Y.-F. Tseng, Enabled/disabled predicate encryption in clouds, Future Generation Computer Systems 62 (2016) 148–160. doi:https://doi.org/10.1016/j.future.2015.12.008.

URL https://www.sciencedirect.com/science/article/pii/S0167739X15003921

[13] J. Lai, X. Zhou, R. Deng, X. Li, K. Chen, Expressive search on encrypted data, in: Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security, 2013, pp. 243–252.

[14] A. Guillevic, Comparing the pairing efficiency over composite-order and prime-order elliptic curves, in: Applied Cryptography and Network Security, 2013, pp. 357–372.

[15] M. H. Ameri, M. Delavar, J. Mohajeri, M. Salmasizadeh, A key-policy attribute-based temporary keyword search scheme for secure cloud storage, IEEE Transactions on Cloud Computing (2018) 1–1.

[16] Y. Miao, J. Ma, X. Liu, X. Li, Q. Jiang, J. Zhang, Attribute-based keyword search over hierarchical data in cloud computing, IEEE Transactions on Services Computing (2018) 1–1.

[17] Y. Miao, J. Ma, X. Liu, J. Weng, H. Li, Lightweight fine-grained search over encrypted data in fog computing, IEEE Transactions on Services Computing (2018) 1–1.

[18] H. Wang, X. Dong, Z. Cao, Multi-value-independent ciphertext-policy attribute based encryption with fast keyword search, IEEE Transactions on Services Computing (2018) 1–1.

[19] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, B. Waters, Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption, in: Advances in Cryptology – EUROCRYPT 2010, 2010, pp. 62–91.

[20] H. Fei, Q. Jing, Z. Huawei, H. Jiankun, A general transformation from KP-ABE to searchable encryption, in: Cyberspace Safety and Security, Springer Berlin Heidelberg, 2012, pp. 165–178.

[21] F. Han, J. Qin, H. Zhao, J. Hu, A general transformation from KP-ABE to searchable encryption, Future Generation Computer Systems 30 (2014) 107 – 115, special Issue on Extreme Scale Parallel Architectures and Systems, Cryptography in Cloud Computing and Recent Advances in Parallel and Distributed Systems, ICPADS 2012 Selected Papers. doi:https://doi.org/10.1016/j.future.2013.09.013.

[22] C. Shen, Y. Lu, J. Li, Expressive public-key encryption with keyword search: Generic construction from KP-ABE and an efficient scheme over prime-order groups, IEEE Access 8 (2020) 93–103. doi:10.1109/ACCESS.2019.2961633.

[23] Z. Lv, C. Hong, M. Zhang, D. Feng, Expressive and secure searchable encryption in the public key setting, in: Information Security, 2014, pp. 364–376.

[24] Q. Zheng, S. Xu, G. Ateniese, VABKS: Verifiable attribute-based keyword search over outsourced encrypted data, in: IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, 2014, pp. 522–530.

[25] H. Cui, Z. Wan, R. H. Deng, G. Wang, Y. Li, Efficient and expressive keyword search over encrypted data in cloud, IEEE Transactions on Dependable and Secure Computing 15 (3) (2018) 409–422.

[26] R. Meng, Y. Zhou, J. Ning, K. Liang, J. Han, W. Susilo, An efficient key-policy attribute-based searchable encryption in prime-order groups, in: Provable Security, Cham, 2017, pp. 39–56.

[27] J. Hao, J. Liu, H. Wang, L. Liu, M. Xian, X. Shen, Efficient attribute-based access control with authorized search in cloud storage, IEEE Access (2019) 1–1doi:10.1109/ACCESS.2019.2906726.

[28] S. Chatterjee, A. Menezes, On cryptographic protocols employing asymmetric pairings - the role of ψ revisited, Discrete Applied Mathematics

159 (13) (2011) 1311 – 1322. doi:https://doi.org/10.1016/j.dam.2011.04.021.

[29] A. Beimel, Secure schemes for secret sharing and key distribution, Technion-Israel Institute of technology, Faculty of computer science, 1996.

[30] D. Boneh, A. Raghunathan, G. Segev, Function-private identity-based encryption: Hiding the function in functional encryption, in: R. Canetti, J. A. Garay (Eds.), Advances in Cryptology – CRYPTO 2013, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 461–478.

[31] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: Proceedings of the 13th ACM conference on Computer and communications security, 2006, pp. 89–98.

[32] Y.-F. Tseng, C.-I. Fan, C.-W. Lin, Provably secure ciphertext-policy attribute-based encryption from identity-based encryption, Journal of Universal Computer Science 25 (3) (2019) 182–202.

[33] C.-I. Fan, Y.-F. Tseng, J.-J. Huang, S.-F. Chen, H. Kikuchi, Multireceiver predicate encryption for online social networks, IEEE Transactions on Signal and Information Processing over Networks 3 (2) (2017) 388–403. doi:10.1109/TSIPN.2017.2697580.

[34] S. Hohenberger, B. Waters, Attribute-based encryption with fast decryption, in: K. Kurosawa, G. Hanaoka (Eds.), Public-Key Cryptography – PKC 2013, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 162–179.

[35] A. Miyaji, M. Nakabayashi, S. Takano, New explicit conditions of elliptic curve traces fir fr-reuction, IEICE Transactions on Fundamentals of Electronic, Communications and Computer Sciences 84 (5) (2001) 1234–1243.

[36] A. Lewko, B. Waters, Decentralizing attribute-based encryption, in: K. G. Paterson (Ed.), Advances in Cryptology – EUROCRYPT 2011, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 568–588.

[37] J.-W. Cai, Attribute hiding key-policy attribute-based encryption, Master's thesis, National Taiwan Ocean University, No. 2, Beining Rd., Zhongzheng Dist., Keelung City 20224, Taiwan (R.O.C.) (2013).

[38] Y.-T. Lin, Key-policy attribute based encryption with hidden ciphertext attributes, Master's thesis, National Taiwan Ocean University, No. 2, Beining Rd., Zhongzheng Dist., Keelung City 20224, Taiwan (R.O.C.) (2014).

[39] T. Nishide, K. Yoneyama, K. Ohta, Attribute-based encryption with partially hidden encryptor-specified access structures, in: S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.), Applied Cryptography and Network Security, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 111–129.

[40] J. Li, K. Ren, B. Zhu, Z. Wan, Privacy-aware attribute-based encryption with user accountability, in: P. Samarati, M. Yung, F. Martinelli, C. A. Ardagna (Eds.), Information Security, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 347–362.

[41] J. Lai, R. H. Deng, Y. Li, Expressive CP-ABE with partially hidden access structures, in: Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ASIACCS '12, ACM, New York, NY, USA, 2012, pp. 18–19. doi:10.1145/2414456.2414465.
URL http://doi.acm.org/10.1145/2414456.2414465

[42] H. Cui, R. H. Deng, G. Wu, J. Lai, An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, in: Proceedings of the 10th International Conference on Provable Security - Volume 10005, ProvSec 2016, Springer-Verlag New York, Inc., New York, NY, USA, 2016, pp. 19–38.

[43] Y. Chen, W. Li, F. Gao, W. Yin, K. Liang, H. Zhang, Q. Wen, Efficient Attribute-Based Data Sharing Scheme with Hidden Access Structures, The Computer Journaldoi:10.1093/comjnl/bxz052.

[44] L. Zhang, G. Hu, Y. Mu, F. Rezaeibagha, Hidden ciphertext policy attribute-based encryption with fast decryption for personal health record system, IEEE Access 7 (2019) 33202–33213.

[45] A. Kiayias, K. Samari, Lower bounds for private broadcast encryption, in: M. Kirchner, D. Ghosal (Eds.), Information Hiding, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 176–190.

[46] D. Boneh, M. Zhandry, Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation, in: J. A. Garay, R. Gennaro (Eds.), Advances in Cryptology – CRYPTO 2014, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 480–499.

[47] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, A. D. Rubin, Charm: a framework for rapidly prototyping cryptosystems, Journal of Cryptographic Engineering 3 (2) (2013) 111–128. doi:10.1007/s13389-013-0057-3.
URL http://dx.doi.org/10.1007/s13389-013-0057-3