Future Generation Computer Systems 30 (2014) 107–115

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A general transformation from KP-ABE to searchable encryption

Fei Han [a], Jing Qin [a,*], Huawei Zhao [b], Jiankun Hu [a,c]

[a] School of Mathematics, Shandong University, Jinan, China
[b] School of Computer Science & Technology, Shandong University of Finance and Economics, Jinan, China
[c] School of Engineering and Information Technology, University of New South Wales Defence Force Academy Canberra, Australia

HIGHLIGHTS

- We have proposed a KP-ABE scheme with "attribute privacy" features.
- We have introduced a general transformation from the KP-ABE scheme to the ABEKS scheme.
- We have designed a secure ABEKS scheme by the transformation.
- We have provided the proof of security and consistency of the ABEKS scheme.

ARTICLE INFO

Article history:
Received 31 December 2012
Received in revised form
22 July 2013
Accepted 5 September 2013
Available online 17 September 2013

Keywords:
KP-ABE
PEKS
ABEKS
Searchable encryption
Weak anonymity

ABSTRACT

Users are inclined to share sensitive data in a remote server if no strong security mechanism is in place. Searchable encryption satisfies the need of users to execute a search encrypted data. Previous searchable encryption methods such as "public key encryption with keyword search (PEKS)" restricted the data access to certain users, because only the assigned users were able to search the encrypted data. In this paper we will discuss the relation between Attribute Based Encryption (ABE) and searchable encryption and define a weak anonymity of the ABE scheme, named "attribute privacy". With this weak anonymity, we propose a general transformation from ABE to Attribute Based Encryption with Keyword Search (ABEKS) and a concrete attribute private key-policy ABE (KP-ABE) scheme. We present an ABEKS scheme based on this KP-ABE scheme and permit multi-users to execute a flexible search on the remote encrypted data.

1. Introduction

Users tend to take advantage of the storage and computing resources of a data server such as the public cloud. However, in such an environment data is often stored in a third party's service. How to guarantee the data security attracts great attention. The notion "searchable encryption" is one of the methods to solve the problem. It allows users to securely execute the search operation on the remote data stored in a third party's service. Most previous works, e.g. PEKS, were originally designed for the purpose of intelligent email routing. They assumed that only the mail receiver is permitted to search on the encrypted mails. However, in a general database service, especially in a cloud computing service, data is shared for public users. Hence, it is desirable that searchable encryption should enable multi-users to search on the data. The construction of efficient multi-user searchable encryption is important and challenging.

Searchable encryption is an encryption algorithm with an additional searchable property. Data owners store their encrypted data in a third storage server. Users are allowed to search the encrypted data with proper keywords and get the desired encrypted data. According to the source of encrypted data, the model of searchable encryption can be classified as follows.

(i) The encrypted data is uploaded to the server by searchers and will be searched by the users later. Many searchable encryption schemes have focused on this model, most of them being based on symmetric encryption [1–3].
(ii) The encrypted data is uploaded to the servers, then all the users are allowed to execute a search on the data. Ogata and Kurosawa [4] provided a solution using this model.
(iii) The encrypted data is uploaded to the server by the data owner, and all the public users are allowed to search on the data if they are interested in it. Different from model (ii), here the server will receive the encrypted data from the data owner. Undoubtedly, in current website storage services, especially in cloud computing services, users may share their data with other users so that they can cooperate with others online. Therefore this model is attracting more interest. Our

* Corresponding author.
E-mail addresses: hanf1987@163.com (F. Han), qinjing@sdu.edu.cn (J. Qin), zhuav@163.com (H. Zhao), J.Hu@adfa.edu.au (J. Hu).

0167-739X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.future.2013.09.013

work in this paper will focus on this model. Previous works mostly use the identity based encryption (IBE) scheme to construct a PEKS scheme and the data owner should assign which identities can search on the data. So these schemes have a restriction that only a single user implements the search [5,6]. We will introduce ABE to this model and enable multi-users to search on the same data, where users can utilize a more flexible search policy through the construction of an access control policy.

ABE [7–10] is a powerful encryption algorithm. It decomposes the user identity as a set of attributes, and enables every user to have its unique attribute set. Goyol et al. [8] formulated two forms of the ABE scheme. One is KP-ABE, and the other is cipher-text-policy ABE (CP-ABE). In KP-ABE, keys are associated with the access control policy, and cipher-texts are associated with the attribute set. In CP-ABE, it is just the opposite: keys are associated with attribute set, and cipher-texts are associated with access policy. CP-ABE seems to be more realistic for practical applications than KP-ABE. This is because, in CP-ABE, the encryptor dominates the construction of the access control policy and determines what attributes could pass the access policy. But in KP-ABE, the decryptor defines the access control policy. However, KP-ABE is more suitable for searchable encryption compared with CP-ABE. For the KP-ABE scheme, the data owner uses the keywords of the data files as an attribute set to encrypt the data files, then uploads the cipher-text to the servers. Users construct the access control policy by the keyword set and the search policy, such as $\{(Tom \cup Dinner) \cap Urgent\}$ (i.e. the receiver wants to search for a data file whose keyword is $\{Tom, Urgent\}$ or $\{Dinner, Urgent\}$). Then an attribute authority generates the secret key for the users. And they send the secret key to the server S. If S could decrypt the cipher-text successfully, the users should get the desired data file; otherwise, the search failed. Obviously, in KP-ABE the access control policy was defined by decryptor. So the receiver could construct a complicated and precise search policy to implement a flexible search.

Hierarchical IBE (HIBE) is considered to have a close relation to ABE scheme. Anonymous HIBE was introduced in [11,12]. We have found that their techniques are very useful to construct our attribute-private ABE and ABEKS scheme.

Our contributions. We observed that the ABE scheme needs to be anonymous. Unfortunately an anonymous ABE normally has a great computational overhead. To address this issue, a weak anonymity feature, namely "attribute privacy", is introduced to fulfill the requirement of ABE to ABEKS. The "attribute privacy" only requires a slight modification from a normal ABE scheme, which incurs little computational overhead. We adopted the techniques used in [11] to construct an attribute private ABE scheme. With this weak anonymity, we propose a general transformation from KP-ABE to ABEKS. By this transformation we construct an ABEKS scheme and permit multi-users to execute a flexible search on the remote encrypted data. The paper in this special issue is based on its corresponding conference version [13]. The following extensive modifications have been made. In the revised version, we modified the application in Section 5, added the formal description of the security game of the ABEKS scheme, and provided a formal proof for the keyword security and consistency of ABEKS scheme in Section 4.

Related works. ABE as a functional encryption, was introduced by Sahai and Waters [7], and then followed by many other researchers. Goyal et al. formulated two forms of ABE scheme, and proposed a KP-ABE scheme [8]. Bethencourt et al. proposed the first CP-ABE scheme [9]. Goyal et al. fulfilled the access control structure and the security proof [14]. Many researchers focus on multi-authority ABE to satisfy the actual requirements [15,16]. All these ABE schemes use the access control tree as the access policy, and they are not expressive. Waters [17] proposed a more expressive and efficient ABE scheme by using an access control matrix as the access policy. Hohenberger and Waters proposed an ABE scheme that can imply fast encryption [18], which made ABE more practical. However, all these researches were selectively secure. Lewko et al. proposed a full secure ABE scheme [19], both in KP and CP versions. It adapted dual system encryption [20] to get a fully secure property. Then Lewko and Waters developed a new methodology to obtain a fully secure ABE scheme from a selectively secure system [21].

The first searchable encryption was introduced by Song [3]. It proposed a scheme that enables a user to implement a search on remote encrypted data, which was based on symmetric encryption. However, it was not fully secure and only supported the two-party model. Also many schemes that are secure searchable encryption based on symmetric encryption such as [1,2] have been proposed. But these schemes were still unsuitable for the third party situation. Boneh et al. proposed PEKS scheme [6]. This scheme allowed a user to search on remote data encrypted by a third party. The study of PEKS scheme turned into the mainstream of searchable public key encryption. Abdalla et al. summarized the previous works of PEKS and proposed the definition "consistency" of searchable encryption [5]. They analyzed the relationship between IBE and PEKS, proposed a general transformation from anonymous IBE to PEKS, and constructed a statistical computational PEKS scheme.

Organization. This paper is organized as follows. Preliminaries are given in Section 2. In Section 3 we will discuss the security issues of KP-ABE and ABEKS, and present a general transformation of KP-ABE to ABEKS. In Section 4 we will propose a concrete ABEKS scheme that enables multi-users to implement their search algorithms. Conclusion will be given in Section 5.

2. Preliminaries

This section will introduce the formal definitions of access control structure, linear secret sharing schemes (LSSS), background knowledge of composite order bilinear groups and complexity assumptions. The details can be found in [19].

2.1. Access control structure

Definition 2.1 (*Access Structure [22]*)**.** Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \ldots, P_n\}$, i.e. $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \setminus \{\varnothing\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In our settings, attributes will play the role of parties. We will only deal with monotone access structures.

We now introduce the LSSS definition adapted from [22].

Definition 2.2 (*Linear Secret Sharing Scheme (LSSS)*)**.** A secret sharing scheme $\prod$ over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if

(i) The shares for each party form a vector over $\mathbb{Z}_p$.
(ii) There exists a matrix A called the share-generating matrix for $\prod$. The matrix A has l rows and n columns. For all $i = \{1, \ldots, l\}$, the ith row of A is labeled by a party ρ (i) (ρ is a function from $\{1, \ldots, l\}$ to $\mathcal{P}$). When we consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then Av is the vector of l shares of the secret s according to $\prod$. The share $(Av)_i$ belongs to a party $\rho(i)$.

The linear reconstruction property is described as follows. Assume that $\prod$ is an LSSS for access structure A. Let S be an authorized set, and define $I \subseteq \{1, \ldots, l\}$ as $I = \{i | \rho(i) \in S\}$. Then there exists

constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$, such that for any valid shares $\{\lambda_i\}$ of a secret s according to $\prod$, we will have $\sum_{i\in I} \omega_i\lambda_i = s$. These constants $\{\omega_i\}$ can be found in polynomial time of the size of share-generating matrix A [22]. And for unauthorized sets, no such constants $\{\omega_i\}$ exist.

2.2. Composite order bilinear groups and complexity assumption

The ABEKS scheme we will propose is constructed from a modified KP-ABE scheme of [19]. We briefly introduce the composite order bilinear groups and three complexity assumptions at first. Interested readers are referred to more details in [19].

Composite order bilinear groups. Composite order bilinear groups were first introduced in [23]. We define a group generator $\mathcal{G}$, an algorithm that takes a security parameter λ as input and outputs a description of a bilinear group G. Let $\mathcal{G}$ output $(p_1, p_2, p_3, p_4, G, G_T, e)$, where p_1, p_2, p_3, p_4 are distinct primes, G and G_T are cyclic groups of order $N = p_1p_2p_3p_4$, $G_{p_1}, G_{p_2}, G_{p_3}$ and G_{p_4} denote the subgroup of order p_1, p_2, p_3 and p_4 in G respectively, and $e : G \times G \to G_T$ is a map that satisfies

(i) (Bilinear) $\forall g, h \in G, a, b \in \mathbb{Z}_N, e(g^a, h^b) = e(g, h)^{ab}$.
(ii) (Non-degenerate) $\exists g \in G$ such that $e(g, g)$ has order N in G_T.
(iii) (Orthogonality) $\forall h_i \in G_{p_i}$ and $h_j \in G_{p_j}$, $i \neq j$, $e(h_i, h_j) = I$, where I is the identity element in G_T.

Assumption 1. Given a group generator $\mathcal{G}$, we define the following distribution

$$\mathbb{G} = (N = p_1p_2p_3p_4, G, G_T, e) \xleftarrow{R} \mathcal{G}, \quad g \xleftarrow{R} G_{p_1},$$
$$X_3 \xleftarrow{R} G_{p_3},$$
$$g_4 \xleftarrow{R} G_{p_4}, \quad D = (\mathbb{G}, g, X_3, g_4), \quad T_1 \xleftarrow{R} G_{p_1p_2},$$
$$T_2 \xleftarrow{R} G_{p_1},$$

and the advantage of an algorithm $\mathcal{A}$ in breaking Assumption 1 to be

$$Adv1_{\mathcal{G},\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(D, T_1) = 1] - \Pr[\mathcal{A}(D, T_2) = 1]|.$$

Definition 2.3. We say that $\mathcal{G}$ satisfies Assumption 1 if $Adv1_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any polynomial time algorithm $\mathcal{A}$.

Assumption 2. Given $\mathcal{G}$, we define the following distribution

$$\mathbb{G} = (N = p_1p_2p_3p_4, G, G_T, e) \xleftarrow{R} \mathcal{G},$$
$$g \xleftarrow{R} G_{p_1}, \quad X_2, Y_2 \xleftarrow{R} G_{p_2}, \quad X_3, Y_3 \xleftarrow{R} G_{p_3},$$
$$g_4 \xleftarrow{R} G_{p_4}$$
$$D = (\mathbb{G}, g, g_4, X_3, g^sX_2, Y_2Y_3), \quad T_1 \xleftarrow{R} G_{p_1p_2p_3},$$
$$T_2 \xleftarrow{R} G_{p_1p_3}$$

and the advantage of $\mathcal{A}$ in breaking Assumption 2 to be

$$Adv2_{\mathcal{G},\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(D, T_1) = 1] - \Pr[\mathcal{A}(D, T_2) = 1]|.$$

Definition 2.4. We say that $\mathcal{G}$ satisfies Assumption 2 if $Adv2_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any polynomial time algorithm $\mathcal{A}$.

Assumption 3. Given $\mathcal{G}$, we define the following distribution

$$\mathbb{G} = (N = p_1p_2p_3p_4, G, G_T, e) \xleftarrow{R} \mathcal{G}, \quad \alpha, s \xleftarrow{R} \mathbb{Z}_N,$$
$$g \xleftarrow{R} G_{p_1}, \quad X_2, Y_2, Z_2 \xleftarrow{R} G_{p_2}, \quad X_3 \xleftarrow{R} G_{p_3},$$
$$X_4, g_4 \xleftarrow{R} G_{p_4}$$
$$D = (\mathbb{G}, g, X_3, g_4, g^\alpha X_2, g^sY_2, Z_2),$$
$$T_1 \xleftarrow{R} e(g, g)^{\alpha s}, \quad T_2 \xleftarrow{R} G_T,$$

and the advantage of $\mathcal{A}$ in breaking Assumption 3 to be

$$Adv3_{\mathcal{G},\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(D, T_1) = 1] - \Pr[\mathcal{A}(D, T_2) = 1]|.$$

Definition 2.5. We say that $\mathcal{G}$ satisfies Assumption 3 if $Adv3_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any polynomial time algorithm $\mathcal{A}$.

Here we will give a sketch of the proof for the above three assumptions' hardness. We have adopted the notation of [19] to prove the hardness. For every element in G that can be expressed as $g_{p_1}^{a_1}g_{p_2}^{a_2}g_{p_3}^{a_3}g_{p_4}^{a_4}$, we denote it as (a_1, a_2, a_3, a_4). For every element in G_T such as the form

$$e(g_{p_1}, g_{p_1})^{a_1}e(g_{p_2}, g_{p_2})^{a_2}e(g_{p_3}, g_{p_3})^{a_3}e(g_{p_4}, g_{p_4})^{a_4}$$

is denoted as $[a_1, a_2, a_3, a_4]$. Random variables are denoted as

$$X = (X_1, Y_1, Z_1, W_1), \quad Y = (X_1, Y_2, Z_2, W_2) \in G.$$

And in this case X and Y share the same element in the G_{p_1} subgroup. Define $L := \{i|e(T_0, A_i) \neq e(T_1, A_i)\}$. Using the method above, we can express the parameters used in the assumptions as the vector form, and prove the assumptions on the basis of Theorems 34 and 35 in [19].

For Assumption 1, we denote the parameters (g, X_3, g_4) as follows:

$$A_1 = (1, 0, 0, 0), \quad A_2 = (0, 0, 1, 0), \quad A_3 = (0, 0, 0, 1),$$
$$T_0 = (S, Y_2, 0, 0), \quad T_1 = (S, 0, 0, 0).$$

It is clear that T_0 and T_1 are both independent of $\{A_1, \ldots, A_3\}$, since S is unrelated to $\{A_1, \ldots, A_3\}$. Thus, assuming the factorization of n is hard, then our assumption is hard.

For Assumption 2, we denote the parameters $(g, X_3, g_4, g^sX_2, Y_2Y_3)$ as follows:

$$A_1 = (1, 0, 0, 0), \quad A_2 = (0, 0, 1, 0),$$
$$A_3 = (0, 0, 0, 1), \quad A_4 = (S, 1, 0, 0),$$
$$A_5 = (0, X_2, X_3, 0).$$
$$T_0 = (Z_1, Z_2, Z_3, 0), \quad T_1 = (Z_1, 0, Z_3, 0).$$

We note that $L = \{4, 5\}$. We need to prove the independence. First we need to prove that $e(T_0, A_4) = (SZ_1, Z_2, 0, 0)$ is independent of $F_{0,4} = \{e(A_i, A_j)\} \cup \{e(T_0, A_i)\}_{i\neq 4}$. This claim is true because we cannot get SZ_1 by the combination of $\{e(A_i, A_j)\} \cup \{e(T_0, A_i)\}_{i\neq 4}$. Similarly we can prove the independence between $e(T_1, A_4)$, $e(T_0, A_5)$, $e(T_1, A_5)$ and their corresponding F. Thus our assumption is hard based on the hardness of factorizing n.

For Assumption 3, we denote the parameters $(g, X_3, g_4, g^\alpha X_2, g^sY_2, Z_2)$ as follows:

$$A_1 = (1, 0, 0, 0), \quad A_2 = (0, 0, 1, 0),$$
$$A_3 = (0, 0, 0, 1), \quad A_4 = (B, 1, 0, 0),$$
$$A_5 = (S, X_2, 0, 0), \quad A_6 = (0, Y_2, 0, 0),$$
$$T_0 = (BS, 0, 0, 0), \quad T_1 = (Z_1, Z_2, Z_3, 0).$$

Clearly T_1 is independent of $\{A_1, \ldots, A_6\}$ because Z_1, Z_2, Z_3 is unrelated with $\{A_1, \ldots, A_6\}$. The term BS of T_0 can only be constructed by computing $e(A_4, A_5)$. But this will result in an additional term X_2 that cannot be canceled. Hence T_0 is also independent of $\{A_1, \ldots, A_6\}$. Thus our assumption is hard based on the hardness of factorizing n.

3. ABE and ABEKS

Now we will introduce new formulations of attribute private KP-ABE and ABEKS, then give a general transformation from ABE to ABEKS.

3.1. Attribute based encryption

A key-policy attribute based encryption consists of four algorithms: **Setup**, **Enc**, **KeyGen**, **Dec**.

Setup $(\lambda, U) \rightarrow (PK, MSK)$.

This algorithm takes the security parameter λ and attributes universe description U as inputs, and outputs the public parameter PK and a master secret key MSK.

Enc $(M, PK, H, S) \rightarrow CT$.

This algorithm takes the message M, the public parameter PK, and a set of attribute S, a hash function H, then outputs a ciphertext CT.

KeyGen $((\mathbb{A}, \rho), MSK, PK) \rightarrow SK$.

This algorithm takes an access structure $\mathbb{A}$, the master secret key MSK and the public parameter PK as inputs, and outputs a secret key SK.

Dec $(CT, PK, SK) \rightarrow M$.

This algorithm takes a ciphertext CT encrypted under hashed attributes set $H(S)$ (where $H(S) = \{H(i)|i \in S\}$) as input, and outputs the message M, if $H(S)$ satisfies the access structure $\mathbb{A}$.

Data privacy.

Data Privacy for an ABE scheme means that an adversary cannot distinguish the encryption of two challenge messages of the choices, even if it is given an oracle of the secret key for any access control policy. Let the message space $MsgSp$, an adversary $\mathcal{A}$ and bit $b \in \{0, 1\}$, define the following experiment:

$Exp^{abe\text{-}ind\text{-}cpa\text{-}b}_{\mathcal{ABE},\mathcal{A}}(k)$,

$ACTSet \leftarrow \phi$; $(PK, MSK) \leftarrow_R$ Setup (1^k);

pick Random Oracle H,

$(S, M_0, M_1, state) \leftarrow_R \mathcal{A}^{KeyGen,H}(find, pk)$;

$C \leftarrow_R \text{Enc}(pk, S, M_b)$,

$b' \leftarrow_R \mathcal{A}^{KeyGen,H}(guess, C, state)$,

if $M_0, M_1 \not\subset MsgSp(k)$ or S satisfies acc_i, return $\perp$,

else return b'.

The oracle KeyGen(id) is defined as

ACTSet $\leftarrow ACTSet \cup \{acc_i\}$; $usk_i \leftarrow_R \text{KeyGen}(msk, acc_i)$;

return usk_i.

The advantage of $\mathcal{A}$ in the experiment is defined as

$$Adv^{abe\text{-}ind\text{-}cpa}_{\mathcal{ABE},\mathcal{A}}(k) = |\Pr[Exp^{abe\text{-}ind\text{-}cpa\text{-}1}_{\mathcal{ABE},\mathcal{A}}(k) = 1] - \Pr[Exp^{abe\text{-}ind\text{-}cpa\text{-}0}_{\mathcal{ABE},\mathcal{A}}(k) = 1]|.$$

$\mathcal{ABE}$ is an ABE-IND-CPA-secure if the advantage function is negligible for all polynomial time algorithms (PTAs) $\mathcal{A}$.

Anonymity.

The anonymity of ABE was defined as the above experiment. However, an anonymous ABE brought more complicated computing overhead. We give a little weak definition of anonymity named as "attribute privacy".

Attribute privacy.

Assume that an adversary $\mathcal{A}$ is given a secret key generator as an oracle, then it chooses two challenge attribute sets S_0, S_1 and two challenge messages M_0, M_1. The challenger encrypts S_b with M_b, and $\mathcal{A}$ cannot obtain any information on b.

$Exp^{abe\text{-}attp\text{-}cpa\text{-}b}_{\mathcal{ABE},\mathcal{A}}(k)$,

$ACTSet \leftarrow \phi$, $(PK, MSK) \leftarrow_R$ Setup (1^k),

pick Random Oracle H,

$(S_0, S_1, M_0, M_1, state) \leftarrow_R \mathcal{A}^{KeyGen,H}(find, pk)$, $C \leftarrow_R$

Enc(pk, S_b, M_b),

$b' \leftarrow_R \mathcal{A}^{KeyGen,H}(guess, C, state)$,

if $M_0, M_1 \not\subset MsgSp(k)$ or S_b satisfies acc_i, return $\perp$, else return b'.

The oracle KeyGen(id) is defined as

ACTSet $\leftarrow ACTSet \cup \{acc_i\}$, $usk_i \leftarrow_R \text{KeyGen}(msk, acc_i)$,

return usk_i.

The advantage of $\mathcal{A}$ in the experiment is defined as

$$Adv^{abe\text{-}attp\text{-}cpa}_{\mathcal{ABE},\mathcal{A}}(k) = |\Pr[Exp^{abe\text{-}attp\text{-}cpa\text{-}1}_{\mathcal{ABE},\mathcal{A}}(k) = 1] - \Pr[Exp^{abe\text{-}attp\text{-}cpa\text{-}0}_{\mathcal{ABE},\mathcal{A}}(k) = 1]|.$$

$\mathcal{ABE}$ is an ABE-attp-CPA-secure if the advantage function is negligible for all PTAs $\mathcal{A}$.

3.2. Attribute based encryption with keyword search

ABEKS employs the brilliant access control property from KP-ABE, and searchers can define a flexible search policy by constructing an access policy. We utilize the encrypted data file as a ABEKS plaintext. The encryptor encrypts the plaintext with the keyword set of the data file, the searchers construct an access policy to get a secret key as a trapdoor, and servers decrypt the ciphertext with the trapdoor to determine whether the data file is desired.

An ABEKS scheme consists of five algorithms: **Setup**, **Enc**, **KeyGen**, **TrapDoor**, **Test**.

Setup $(\lambda, U) \rightarrow (PK, MSK)$.

This algorithm takes the security parameter λ and the attributes universe description U as inputs, and outputs the public parameter PK and the master secret key MSK.

Enc $(\tilde{M}, PK, H(S)) \rightarrow CT$.

This algorithm takes the data message $\tilde{M}(\tilde{M} = C \parallel 0^l)$ as input, uses the public parameter PK and the hashed attributes set $H(S)(H(S) = \{H(i)|\forall i \in S\})$ to encrypt $\tilde{M}$, and outputs the ciphertext CT.

KeyGen $((\mathbb{A}_i, \rho_i), MSK, PK) \rightarrow SK_i$.

This algorithm takes the access control policy $(\mathbb{A}_i, \rho_i)$, the master secret key MSK and the public parameter PK as inputs, outputs the secret key SK_i for user's attribute set.

TrapDoor $((\mathbb{A}_i, \rho_i)) \rightarrow SK_i$.

This algorithm takes the secret key SK_i from KeyGen algorithm as the trapdoor of the keywords for the access control policy $\mathbb{A}_i$.

Test $(CT, PK, SK_i) \rightarrow \tilde{M}$.

This algorithm uses the PK and SK_i to decrypt the ciphertext CT, then gets $\tilde{M}$.

Consistency.

Consistency was first introduced in [5], and it is a crucial security requirement. To discuss the definition we associate the following experiment of an adversary $\mathcal{U}$.

$Exp^{abeks\text{-}consist}_{\mathcal{ABEKS},\mathcal{U}}(k)$,

$(PK, MSK) \leftarrow \text{Setup}(1^k)$, pick random oracle H,

$(w, w') \leftarrow_R \mathcal{U}^H(pk)$, $C \leftarrow_R \mathcal{ABEKS}^H(pk, w)$,

$t_{w'} \leftarrow \text{TrapDoor}^H(msk, w')$.

Test$(C, t_{w'}) = 1$, then $\mathcal{U}$ wins the game.

The advantage of $\mathcal{U}$ is defined as

$$Adv^{abeks\text{-}consist}_{\mathcal{ABEKS},\mathcal{U}}(k) = \Pr[Exp^{abeks\text{-}consist}_{\mathcal{ABEKS},\mathcal{U}}(k) = 1].$$

Abdalla et al. [5] defined three distinct definitions of consistency according to the power of an adversary. A searchable encryption is perfectly consistent if this advantage is 0 for all (computational unrestricted) adversaries $\mathcal{U}$; is statistical consistent if it is negligible for all (computational unrestricted) adversaries $\mathcal{U}$; and is computational consistent if it is negligible for all PTAS $\mathcal{U}$. Our ABEKS scheme satisfies the definition of computational consistent.

Keyword privacy.

The keyword privacy of the ABEKS scheme requires that no adversary can distinguish the encryption of two challenge keywords by its choice, even if it can query the oracle of the trapdoor for any non-challenge keywords. To define the privacy we associate the following experiment with an adversary $\mathcal{A}$ and a bit $b \in \{0, 1\}$

$Exp^{abeks\text{-}ind\text{-}cpa\text{-}b}_{\mathcal{ABEKS},\mathcal{A}}(k)$,

$WSet \leftarrow \phi$, $(PK, MSK) \leftarrow_R \text{Setup}(1^k)$, pick Random Oracle H,

$(w_0, w_1, M_0, M_1, state) \leftarrow_R \mathcal{A}^{TRAPD(),H}(find, pk)$,

$C \leftarrow_R \mathcal{ABEKS}^H(pk, w_b)$,

$b' \leftarrow_R \mathcal{A}^{TRAPD(),H}(guess, H, state)$,

if $b = b'$, $\mathcal{A}$ wins the game, return 1.

The oracle $TRAPD()$ is defined as

$ACTSet \leftarrow ACTSet \cup \{acc\}$, $t_w \leftarrow_R TrapDoor^H(sk, acc)$,

return t_w.

The advantage of $\mathcal{A}$ is defined as

$$Adv^{abeks\text{-}ind\text{-}cpa}_{\mathcal{ABEKS},\mathcal{A}}(k) = |\Pr[Exp^{abeks\text{-}ind\text{-}cpa\text{-}1}_{\mathcal{ABEKS},\mathcal{A}}(k) = 1] - \Pr[Exp^{abeks\text{-}ind\text{-}cpa\text{-}0}_{\mathcal{ABEKS},\mathcal{A}}(k) = 1]|.$$

$\mathcal{ABEKS}$ is an ABEKS-IND-CPA-secure if the advantage function is negligible for all PTAs $\mathcal{A}$.

3.3. The transformation from ABE to ABEKS

Abdalla et al. have introduced the transformation from IBE to PEKS [5]. We expect that there exists a transformation from the ABE scheme to the ABEKS scheme. We will discuss this problem and propose the general transformation from ABE to ABEKS.

Let an ABE-to-ABEKS transformation takes an ABE scheme

$\mathcal{ABE} = \{\text{Setup, Enc, KeyGen, Dec}\}$

as input, and output an ABEKS scheme

$\mathcal{ABEKS} = \{\text{Setup, Enc, KeyGen, Trapdoor, Dec}\}$.

The transformation is designed as follows.

$\text{Setup}_{\mathcal{ABEKS}}(\lambda, U) = \text{Setup}_{\mathcal{ABE}}(\lambda, U)$,

$\text{Enc}_{\mathcal{ABEKS}}(M, PK, H(S)) = \text{Enc}_{\mathcal{ABE}}(M, PK, H(S))$,

$\text{KeyGen}_{\mathcal{ABEKS}}((\mathbb{A}, \rho), MSK, PK) = \text{KeyGen}_{\mathcal{ABE}}((\mathbb{A}, \rho), MSK, PK)$,

$\text{TrapDoor}(\mathbb{A}, \rho) = \text{KeyGen}_{\mathcal{ABE}}((\mathbb{A}, \rho), MSK, PK)$,

$\text{Test}(CT, PK, SK) = \text{Dec}(CT, PK, SK)$.

From the transformation, we get the consistency of the ABEKS scheme by the data privacy of ABE, while the keyword privacy of ABEKS comes from the attribute privacy of the ABE scheme.

Theorem 3.1. *Suppose $\mathcal{ABE}$ is an ABE scheme and $\mathcal{ABEKS}$ is an ABEKS scheme derived from ABE-to-ABEKS. If $\mathcal{ABE}$ is fully secure, then $\mathcal{ABEKS}$ is computational consistent. If $\mathcal{ABE}$ is attribute private, then $\mathcal{ABEKS}$ is keyword private.*

Proof. Let $\mathcal{U}$ be a PTA attacking the computational consistency of $\mathcal{ABEKS}$, an adversary $\mathcal{A}$ attacking the ABE-IND-CPA-secure of $\mathcal{ABE}$. In the *find* stage, $\mathcal{A}$ gets challenge keyword set W, W' by running $\mathcal{U}(pk)$, then returns W as challenge attribute set and picks $M_0, M_1 \leftarrow_R \{0, 1\}^k$ as challenge messages. In the *guess* stage, $\mathcal{A}$ is given C encrypted from $R_{b,b\in\{0,1\}}$ with W, then it uses the *KeyGen* oracle to obtain the trapdoor $t_{W'}$ of W'. If $Dec(t_{W'}, C) = R_1$, then it returns 1 else returns 0. Obviously,

$\Pr[Exp^{abe\text{-}ind\text{-}cpa\text{-}1}_{\mathcal{ABE},\mathcal{A}}(k) = 1] \geq \Pr[Exp^{abeks\text{-}consist}_{\mathcal{ABEKS},\mathcal{U}}(k) = 1]$,

$\Pr[Exp^{abe\text{-}ind\text{-}cpa\text{-}0}_{\mathcal{ABE},\mathcal{A}}(k) = 1] \leq 2^{-k}$.

So $Adv^{abeks\text{-}consist}_{\mathcal{ABEKS},\mathcal{U}}(k) \leq Adv^{abe\text{-}ind\text{-}cpa}_{\mathcal{ABE},\mathcal{A}}(k) + 2^{-k}$. Hence we have proved the first claim.

Let $\mathcal{U}$ be a PTA attacking the attribute privacy of $\mathcal{ABE}$, and an adversary $\mathcal{A}$ attacking the keyword privacy of $\mathcal{ABEKS}$. In the *find* stage, $\mathcal{A}$ gets the challenge keyword set W_0, W_1, and then is given C encrypted by M_b with S_b. $\mathcal{A}$ can run $\mathcal{U}$ to get the information of b and we have

$Adv^{abeks\text{-}ind\text{-}cpa}_{\mathcal{ABEKS},\mathcal{U}}(k) \leq Adv^{abe\text{-}attp\text{-}cpa}_{\mathcal{ABE},\mathcal{A}}(k)$.

So we have proved the second claim.

4. A concrete ABEKS scheme

Firstly we modify the KP-ABE in [19] to get a weak anonymous KP-ABE scheme with the techniques used in [11], then we construct our ABEKS scheme through the transformation given in Section 3.3.

4.1. Key-policy attribute based encryption

We pick a hash function H to compute each attribute in attributes set U and introduce additional elements in the G_{p_4} subgroup to get the weak anonymity property.

Setup $(\lambda, U) \rightarrow (PK, MSK)$.

First, the algorithm chooses a bilinear group G of order $p_1p_2p_3p_4$, then picks up random numbers $\alpha \in \mathbb{Z}_N, g, X_1 \in G_{p_1}$, where G_{p_1} is the subgroup of order p_1 in G. For any attribute i in the global universe attribute set U, the algorithm picks up a hash function H, and computes $H(i)$, then chooses a random number $s_{H(i)} \in \mathbb{Z}_N, X_3, g_4$ as the generators of $G_{p_3}, G_{p_4}, X_4 \in G_{p_4}$, $t = X_1X_4$. We define

$PK = \{N, g, g_4, e(g, g)^\alpha, H, t, T_{H(i)} = g^{s_{H(i)}}, \ \forall i\}$,

$MSK = \{X_1, X_3, \alpha\}$.

Enc $(M, PK, H, S) \rightarrow CT$.

This algorithm picks up a random $s \in \mathbb{Z}_N, R, R' \in G_{p_4}$, and computes $H(S) = \{H(i) | i \in S\}$ for any attribute $i \in S$. The ciphertext is given as

$CT = \{C = Me(g, g)^{\alpha s}, C_0 = g^sR, C_{H(i)} = (tT_{H(i)})^sR', \ \forall i \in S\}$,

which also includes the hashed attributed set $H(S)$.

KeyGen $((\mathbb{A}, \rho), MSK, PK, H) \rightarrow SK$.

$\mathbb{A}$ is a matrix, A_x is the xth row of $\mathbb{A}$, ρ is a map and $\rho : A_x \rightarrow \rho(x) \in H(S)$. This algorithm picks up a random vector u such that the first term of u is α and the other terms are random numbers. For each A_x, it chooses random numbers $r_x \in \mathbb{Z}_N, W_x, V_x \in G_{p_3}$, and the secret key SK is given as:

$$K^1_x = g^{A_xu}(X_1T_{\rho(x)})^{r_x}W_x, \qquad K^2_x = g^{r_x}V_x.$$

Dec $(CT, PK, SK) \rightarrow M$.

Let $H(S)$ denote the hashed attribute set of CT, and $(\mathbb{A}, \rho)$ denote the matrix and row mapping associated with SK. If $H(S)$ satisfies $\mathbb{A}$, then the algorithm finds a constants ω_x, such that $\sum_{\rho(x)\in H(S)} \omega_xA_x = 1$ (1 represents the vector of the first term is 1, and others are 0). Compute

$$\prod_{\rho(x)\in H(S)} \frac{e(C_0, K^1_x)^{\omega_x}}{e(C_{\rho(x)}, K^2_x)^{\omega_x}} = \prod_{\rho(x)\in H(S)} \frac{e(g, g)^{s\omega_xA_xu}e(g, X_1T_{\rho(x)})^{s\omega_xr_x}}{e(g, X_1T_{\rho(x)})^{s\omega_xr_x}}$$

$$= e(g, g)^{s\sum_{\rho(x)\in H(S)}\omega_xA_xu} = e(g, g)^{\alpha s}$$

the message can be recovered by $C/e(g, g)^{\alpha s}$.

4.2. Security proof

4.2.1. Security model for attribute-private ABE

Here we will give the security definition for ABE scheme to prove the attribute privacy (the security can be proved by using

a similar method). The following is a security game between a challenger and an adversary.

Setup: The challenger runs the Setup algorithm and gives the public parameters *PK* to the adversary.

Phase 1: The adversary queries the challenger for private keys corresponding to access policies $acc_1, \ldots, acc_{q_1}$.

Challenge: The adversary declares two equal length message M_0 and M_1, and attributes sets S_0 and S_1 that do not satisfy any of the private keys queried above. The challenger flips a random coin $\beta \in \{0, 1\}$, and encrypts M_β with S_β.

Phase 2: The adversary queries the challenger for private keys corresponding to the access policy with the restriction of none of these satisfy S_0 and S_1.

Guess: The adversary outputs a guess β' for β.

4.2.2. Security proof for ABE

We adopt the proof method of dual system encryption [20] which is viewed as a powerful tool for achieving the full security of the ABE scheme. In composite order bilinear groups, the different subgroups of G play different roles in the cryptosystem. The normal encryption operation occurs in the subgroup G_{p_1}, the subgroup G_{p_2} is a semi-functional space, and is only used in the proof system, the subgroup G_{p_3} is applied to randomize the keys, and the elements of the subgroup G_{p_4} are used to guarantee the anonymity of cipher-texts. Keys and cipher-texts will be semi-functional if they contain elements of the G_{p_2} subgroup. Due to the orthogonality, when we pair normal keys with semi-functional cipher-texts or pair semi-functional keys with normal cipher-texts, the G_{p_2} part of semi-functional space will be canceled. When we pair a semi-functional key with a semi-functional cipher-text, the elements of G_{p_2} will not be canceled, thereby the decryption fails unless the extra term happens to be zero. If this cancellation occurs and decryption succeeds, this key is called nominally semi-functional. That is to say, nominally semi-functional keys do not impede the decryption algorithm.

First we need to define semi-functional keys and cipher-texts.

Semi-functional cipher-text. A semi-functional cipher-text is formed as follows. Let g_2 be a generator of G_{p_2}, $R, R' \in G_{p_4}$, c be a random exponent, z_i be random values, and S_β be the challenge attributes set. Then we have

$$C_0 = g^s g_2^c R, \qquad C_{H(i_\beta)} = (tT_{H(i_\beta)})^s g_2^{c \cdot z_{H(i_\beta)}} R', \quad \forall i_\beta \in S_\beta.$$

Semi-functional key. A semi-functional key will be one of two forms below. A semi-functional key of type 1 is formed as follows. A random vector u_2 is chosen (so $u_2 \cdot 1$ is also random, 1 represents the vector of the first term is 1, and others are 0), and then set $\delta_x = A_x \cdot u_2$. Additionally, random exponents γ_x are chosen. The key is defined as:

$$K_x^1 = g^{A_x \cdot u}(X_1 T_{\rho(x)})^{r_x} W_x g_2^{\delta_x + \gamma_x z_{\rho(x)}}, \qquad K_x^2 = g^{\gamma_x} V_x g_2^{\gamma_x}.$$

A semi-functional key of type 2 is very similar to that of type 1, except without the terms $g_2^{\gamma_x z_{\rho(x)}}$ and $g_2^{\gamma_x}$ (i.e. the term γ_x is equal to zero):

$$K_x^1 = g^{A_x \cdot u}(X_1 T_{\rho(x)})^{r_x} W_x g_2^{\delta_x}, \qquad K_x^2 = g^{\gamma_x} V_x.$$

Note that when using a semi-functional key to decrypt a semi-functional cipher-text, we can get an additional term:

$$e(g_2, g_2)^{\sum_{\rho(x) \in S} c \omega_x \delta_x} = e(g_2, g_2)^{c u_2 \cdot 1}.$$

We also note that the values z_i used in semi-functional cipher-texts and semi-functional keys of type 1 are the same. We call a semi-functional key a nominally semi-functional key if $u_2 \cdot 1 = 0$ (1 represents the vector of the first term is 1, and others are 0). Notice that a nominally semi-functional key can decrypt a semi-functional cipher-text successfully.

We will prove the adaptive security of our system from our assumptions using a sequence of games. The first game, Game_{real}, is the real security game (the challenge cipher-text and all keys are normal). In Game_0, all keys are normal, but the challenge cipher-text is semi-functional. We let q be the number of key queries made by the adversary. For k from 1 to q, we define:

$\text{Game}_{k,1}$. In this game, the challenge cipher-text is semi-functional, the first $k-1$ keys are semi-functional of type 2, the kth key is semi-functional of type 1, and the remaining keys are normal.

$\text{Game}_{k,2}$. In this game, the challenge cipher-text is semi-functional, the first $k-1$ keys are semi-functional of type 2, and the remaining keys are normal. (Note that $\text{Game}_{0,2}$ is another notation of Game 0 and $\text{Game}_{0,1}$.)

Notice that in $\text{Game}_{q,2}$, all keys are semi-functional of type 2.

In the final game Game_{final}, all keys are semi-functional of type 2 and the ciphertext is a semi-functional encryption of a random message, which are independent of the two challenge messages. Obviously in Game_{final} the adversary's advantage is 0. We need to prove these games are distinguishable. This is done by the following lemmas.

Lemma 1. *Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $\text{Game}_{real}\ Adv_{\mathcal{A}} - \text{Game}_0\ Adv_{\mathcal{A}} = \varepsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 1.*

Lemma 2. *Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $\text{Game}_{k-1,2} Adv_{\mathcal{A}} - \text{Game}_{k,1} Adv_{\mathcal{A}} = \varepsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 2.*

Lemma 3. *Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $\text{Game}_{k,1} Adv_{\mathcal{A}} - \text{Game}_{k,2} Adv_{\mathcal{A}} = \varepsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 2.*

Lemma 4. *Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $\text{Game}_{q,2}\ Adv_{\mathcal{A}} - \text{Game}_{final}\ Adv_{\mathcal{A}} = \varepsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ε in breaking Assumption 3.*

The proof sketch of our ABE scheme is similar to the security proof of KP-ABE in [19] except that we added random elements of G_{p_4} to the cipher-text and keys. The challenge cipher-text is given as

$$C_0 = g^s g_2^c R, \qquad C_{H(i_\beta)} = (tT_{H(i_\beta)})^s g_2^{c \cdot z_{H(i_\beta)}} R', \quad \forall i_\beta \in S_\beta$$

which is determined by the security model of the ABE scheme. Therefore what we need to do is to prove the above lemmas based on this challenge cipher-text. We can prove the security of the ABE scheme by using a similar method to that stated in [19].

Proof of Lemma 1. Given $\{g, X_1 = g^l, X_3, g_4, t^s (t = X_1 g_4), T\}$ (l, s is randomly chosen), $\mathcal{B}$ will simulate either Game_{real} or Game_0 with $\mathcal{A}$. $\mathcal{B}$ sends $\mathcal{A}$ the public parameters which are formed by choosing random exponents α, s_i. $\mathcal{B}$ can answer the key requests by running the usual key generation algorithm to make normal keys because it knows the *MSK*.

To form the challenge cipher-text M_β for the challenge attributes set S_β, $\mathcal{B}$ choose random elements $R = g_4^a, R' = g_4^b, a, b \in \mathbb{Z}_N$ and implicitly sets s, so that g^s is the G_{p_1} part of T (T is the product of g^s and possibly an element of G_{p_2}):

$$C = M_\beta e(g, g)^{s\alpha} = M_\beta e(g, T)^\alpha, \qquad C_0 = TR,$$

$$C_{H(i_\beta)} = t^s (T)^{s_{H(i_\beta)}} R', \quad \forall i_\beta \in S_\beta.$$

We note that we set $z_i = s_i$, $\forall i$. However the value of z_i modulo p_1 is uncorrelated with the values of s_i modulo p_2 by the Chinese Remainder theorem.

If $T = g^s$, this is a properly distributed normal ciphertext. If $T = g^s X_2$(for $X_2 \in G_{p_2}$), this is a properly distributed semi-functional ciphertext. Thus, $\mathcal{B}$ can break Assumption 1 with advantage ε by simulating with $\mathcal{A}$.

Proof of Lemma 2. Given $\{g, X_3, g_4, g^s X_2, Y_2 Y_3, T\}$, $\mathcal{B}$ will simulate $\text{Game}_{k-1,2}$ or $\text{Game}_{k,1}$. At first, $\mathcal{B}$ sends $\mathcal{A}$ the public parameters $\{N, g, g_4, X_3, e(g,g)^\alpha, t^s(t = X_1 g_4), g^{s_{H(i)}}\}$ ($X_1 = g^l$, l, s is randomly chosen), where it randomly chooses α, values $s_{H(i)}$ and $R = g_4^a$, $R' = g_4^b$, $a, b \in \mathbb{Z}_N$.

To form the challenge cipher-text M_β for the challenge attributes set S_β, $\mathcal{B}$ sets:

$$C = M_\beta e(g^s X_2, g)^\alpha, \qquad C_0 = g^s X_2 R,$$
$$C_{H(i_\beta)} = t^s (g^s X_2)^{s_{H(i_\beta)}} R', \quad \forall i_\beta \in S_\beta.$$

Note that we set $z_i = s_i$, $\forall i$. And also z_i modulo p_1 is uncorrelated with s_i modulo p_2.

Next, we need to form the key queried by adversary. Since $\mathcal{B}$ knows MSK, it can form normal keys for queries $>k$ via the regular key generation algorithm. To create semi-functional keys of type 2 for queries $<k$, $\mathcal{B}$ can randomly choose a vector $\mathbf{u}$ such that $\mathbf{u} \cdot 1 = \alpha$, a vector $\mathbf{u}_2'$, values r_x, X_1 is the G_{p1} of t and group elements W_x, $V_x \in G_{p_3}$. The semi-functional key of type 2 can be defined as:

$$K_x^1 = g^{A_x \cdot u} (X_1 T_{\rho(x)})^{r_x} W_x (Y_2 Y_3)^{A_x \cdot \mathbf{u}_2'}$$
$$K_x^2 = g^{r_x} V_x.$$

We note that $Y_2 = g_2^c$, and $\mathbf{u}_2$ in our description of semi-functional keys above now corresponds to $\mathbf{u}_2 = c \cdot \mathbf{u}_2'$.

To form the kth key, $\mathcal{B}$ will choose to make a normal or nominally semi-functional key of type 1 according to the value of T. Later we will show that a semi-functional key of type 1 is indistinguishable from a nominally one in the view of an adversary.

To form the kth key for (A, ρ), $\mathcal{B}$ randomly chooses a vector $\mathbf{u}_2$ such that $\mathbf{u}_2 \cdot 1 = 0$ and a vector $\mathbf{u}'$ such that $\mathbf{u}' \cdot 1 = \alpha$, and then sets $\mathbf{u} = r\mathbf{u}_2 + \mathbf{u}'$, where g^r is the G_{p_1} part of T. Values γ_x, W_x, V_x are chosen randomly. Hence $\mathcal{B}$ sets the key as:

$$K_x^1 = g^{A_x \cdot \mathbf{u}'} (T)^{A_x \cdot \mathbf{u}_2} (X_1^r T^{z_{\rho(x)}})^{\gamma_x} W_x$$
$$K_x^2 = T^{\gamma_x} V_x.$$

Here we set $r_x = r \cdot \gamma_x$, and r_x modulo p_1 is uncorrelated with γ_x modulo p_2. According to the value of T, this is either a normal key or a nominally semi-functional key of type 1.

Now we need to prove that if the kth key queried by the adversary cannot decrypt a challenger cipher-text, then it will seem properly distributed in the adversary's view. We consider an attribute i that does not belong to the challenge attributes set S_β, then value $z_{H(i)}$ can only appear in the kth key, since any other key is normal or semi-functional of type 2.

So we need here that the kth key cannot decrypt the challenge cipher-text. This means that we need an extra term when computing the bilinear map. So we choose a vector $\mathbf{w}$ such that when computing $\mathbf{w}$ with the corresponding matrix A, we will not get $(1, 0, \ldots, 0)$. We fix a basis containing $\mathbf{w}$, and then set $\mathbf{u}_2 = f\mathbf{w} + \mathbf{u}_2''$ for $f \in \mathbb{Z}_n$ and a uniformly distributed vector $\mathbf{u}_2''$. Here $\mathbf{u}_2''$ leaks no information of f. Obviously $(1, 0, \ldots, 0) \cdot u_2$ is determined by both $\mathbf{u}_2''$ and f. And the computation of matrix with attributes $i \in S_\beta$ only leaks information about $\mathbf{u}_2''$. Then the information of $(1, 0, \ldots, 0) \cdot u_2$ is hidden.

$f\mathbf{w}$ only appears in equation $(T)^{A_x \cdot \mathbf{u}_2} (X_1^r T^{z_{\rho(x)}})^{\gamma_x}$, where $\rho(x)$ is the attribute that will not be appearing in the challenge cipher-text. The information mask of $\gamma_x z_{\rho(x)}$ prevents the adversary from knowing the information about f, unless γ_x modulo p_2 is equal to zero (Obviously this happens with negligible probability). Hence the information shared in the subgroup G_{p_2} is information-theoretically hidden. Thus the cipher-text and kth key are properly distributed in the view of the adversary.

Therefore, $\mathcal{B}$ can break Assumption 2 with advantage ε by simulating $\text{Game}_{k-1,2}$ or $\text{Game}_{k,1}$ with $\mathcal{A}$ according to the value of T.

Proof of Lemma 3. Given $\{g, X_3, g_4, g^s X_2, Y_2 Y_3, T\}$, $\mathcal{B}$ will simulate either $\text{Game}_{k,1}$ or $\text{Game}_{k,2}$ with $\mathcal{A}$. At first, $\mathcal{B}$ sends $\mathcal{A}$ the public parameters $\{N, g, g_4, X_3, e(g,g)^\alpha, t^s(t = X_1 g_4), g^{s_{H(i)}}\}$ ($X_1 = g^l$, l, s is randomly chosen), by choosing α, l, s and the values $s_{H(i)}$ randomly, and then sets $X_1 = g^l$, $z_i = s_i$ for any i.

To form the challenge cipher-text M_β for the challenge attributes set W_β, $\mathcal{B}$ randomly chooses $R = g_4^a$, $R' = g_4^b$, $a, b \in \mathbb{Z}_N$, and sets:

$$C = M_\beta e(g^s X_2, g)^\alpha, \qquad C_0 = g^s X_2 R,$$
$$C_{H(w_{i\beta})} = t^s (g^s X_2)^{s_{H(w_{i\beta})}} R', \quad \forall w_{i\beta} \in W_\beta.$$

Next, we need to form the key queried by the adversary. Since $\mathcal{B}$ knows MSK, it can form normal keys for queries $>k$ via the regular key generation algorithm. To create semi-functional keys of type 2 for queries $<k$, $\mathcal{B}$ can randomly choose a vector $\mathbf{u}$ such that $\mathbf{u} \cdot 1 = \alpha$, a vector $\mathbf{u}_2'$, values r_x, and group elements W_x, $V_x \in G_{p_3}$. The semi-functional key of type 2 can be defined as:

$$K_x^1 = g^{A_x \cdot u} (X_1 T_{\rho(x)})^{r_x} W_x (Y_2 Y_3)^{A_x \cdot \mathbf{u}_2'}$$
$$K_x^2 = g^{\gamma_x} V_x.$$

We note that here $Y_2 = g_2^c$, and $\mathbf{u}_2$ in our description of semi-functional keys above now corresponds to $\mathbf{u}_2 = c \cdot \mathbf{u}_2'$.

For the kth key, $\mathcal{B}$ will choose to form a semi-functional key of type 1 or of type 2, according to the value of T in the assumption. $\mathcal{B}$ first randomly chooses a vector $\mathbf{u}$ such that $\mathbf{u} \cdot 1 = \alpha$ and a vector $\mathbf{u}_2$, values γ_x, elements W_x, $V_x \in G_{p_3}$ and then sets $r_x = r\gamma_x$, where g^r is the G_{p_1} part of T. Hence $\mathcal{B}$ can compute the key as:

$$K_x^1 = g^{A_x \cdot u} (Y_2 Y_3)^{A_x \cdot \mathbf{u}_2'} (X_1^r T^{s_{\rho(x)}})^{\gamma_x} W_x$$
$$K_x^2 = T^{\gamma_x} V_x.$$

If $T \in G_{p_1 p_2 p_3}$, this is a properly distributed semi-functional key of type 1. If $T \in G_{p_1 p_3}$, this is a properly distributed semi-functional key of type 2. Hence $\mathcal{B}$ can break Assumption 2 with advantage ε by simulating with $\mathcal{A}$.

Proof of Lemma 4. Given $\{g, X_3, g_4, g^\alpha X_2, g^s Y_2, Z_2, T\}$, $\mathcal{B}$ will simulate $\text{Game}_{q,2}$ or Game_{final} with $\mathcal{A}$. At first $\mathcal{B}$ sets the public parameters as

$$\{N, X_1 = g^l, g, g_4, X_3, t^s(t = X_1 g_4), e(g,g)^\alpha = e(g, g^\alpha X_2)\}$$

(l, s is randomly chosen).

It then randomly chooses $s_i = z_i$ for any i.

To form the challenge cipher-text M_β for the challenge attributes set S_β, $\mathcal{B}$ randomly chooses $R = g_4^a$, $R' = g_4^b$, $a, b \in \mathbb{Z}_N$ and sets:

$$C = M_\beta T, \qquad C_0 = g^s X_2 R,$$
$$C_{H(i_\beta)} = t^s (g^s X_2)^{s_{H(i_\beta)}} R', \quad \forall i_\beta \in S_\beta.$$

If $T = e(g,g)^{\alpha s}$, this is a semi-functional encryption of M_β. If T is random, this is a semi-functional encryption of a random message and this will leak no information about β to the attacker.

Next we need to form a semi-functional key of type 2 for (A, ρ) (where A has l *columns*), $\mathcal{B}$ chooses $u_1, \ldots, u_{l-1}$ randomly and sets $u_l = \alpha - \sum_{i=1}^{l-1} u_i$ (so that $\mathbf{u} \cdot 1 = \alpha$). $\mathcal{B}$ randomly chooses vector

$\mathbf{u}_2$ of length l, values r_x, elements $W_x, V_x \in G_{p_3}$. Then $\mathcal{B}$ sets the key as:

$$K_x^1 = g^{\sum_{i=1}^{l-1}(A_{x,i}u_i - A_{x,l}u_i)} (g^{\alpha} X_2)^{A_{x,l}} (X_1 T_{\rho(x)})^{r_x} Z_2^{A_x \cdot \mathbf{u}_2}$$
$$= g^{A_x \cdot u} X_1^{r_x} T_{\rho(x)}^{r_x} (Z_2)^{A_x \cdot \mathbf{u}_2} X_2^{A_{x,l}} W_x$$
$$K_x^2 = g^{r_x} V_x$$

where $X_2 = g_2^c, Z_2 = g_2^d$, and in the G_{p_2} subgroup, this will be $g_2^{\delta_x}$ for $\delta_x = A_x \cdot \mathbf{u}_2'$ where $\mathbf{u}_2' = d \cdot \mathbf{u}_2$ except with a value c added in the last coordinate.

This will be a properly distributed semi-functional key of type 2. Hence $\mathcal{B}$ can break Assumption 3 with advantage ε by simulating with $\mathcal{A}$.

Theorem 4.2.1. *If Assumptions 1–3 hold, then our ABE scheme is secure and anonymous.*

Proof. If Assumptions 1–3 hold, by the previous lemma we know that the real security game is indistinguishable from Game_{final}, so β is information-theoretically hidden from the adversary. Hence the adversary cannot gain a non-negligible advantage in breaking the ABE scheme.

4.3. Attribute based encryption with keyword search

We construct our ABEKS scheme by the above KP-ABE scheme. This scheme consists of five algorithms:

Setup $(\lambda, U) \to (PK, MSK)$.

The algorithm chooses a bilinear group G of order $p_1p_2p_3p_4$, picks up random numbers $\alpha \in \mathbb{Z}_N, g \in G_{p_1}$, where G_{p_1} is the subgroup of order p_1 in G. For any attribute i in global universe attributes set U, the algorithm picks up a hash function H, and computes $H(i)$, then chooses a random number $s_{H(i)} \in \mathbb{Z}_N, X_3, g_4$ as the generator of $G_{p_3}, G_{p_4}, X_4 \in G_{p_4}, t = X_1X_4$. We define

$$PK = \{N, g, g_4, e(g,g)^{\alpha}, H, t, T_{H(i)} = g^{s_{H(i)}}, \forall i\},$$
$$MSK = \{X_1, X_3, \alpha\}.$$

Enc $(\tilde{M}, PK, H, W) \to CT$.

$\tilde{M}$ is the encrypted data file $C(\tilde{M} = C \parallel 0^l)$. W is the keyword set of data file. For any keyword $w_i \in W$, compute $H(W) = \{H(w_i) | \forall w_i \in W\}$, where H is the hash function. Then the algorithm picks up a random $s \in \mathbb{Z}_N, R, R' \in G_{p_4}$, and uses the keyword set as an attributes set to encrypt the encrypted data file $\tilde{M}$. The cipher-text is given as:

$$CT = \{C = \tilde{M} e(g,g)^{\alpha s},\ C_0 = g^s R,$$
$$C_{H(w_i)} = (tT_{H(w_i)})^s R',\ \forall w_i \in W\}.$$

KeyGen $((\mathbb{A}_k, \rho_k), MSK, PK) \to SK_k$.

$\mathbb{A}_k$ is the access matrix derived from the keyword which needs to be searched and we have the search policy of user k, $\rho_k : A_{k,x} \to \rho_k(x) \in H(W)$. Then by the *KeyGen* algorithm of above KP-ABE, the algorithm picks up the random numbers $r_{k,x} \in \mathbb{Z}_N, W_{k,x}, V_{k,x} \in G_{p_3}$. The secret key SK_k of user k is given as:

$$K_{k,x}^1 = g^{A_{k,x}u_k} (X_1 T_{\rho_k(x)})^{r_{k,x}} W_{k,x}, \qquad K_{k,x}^2 = g^{r_{k,x}} V_{k,x}.$$

Trapdoor $((\mathbb{A}_k, \rho_k)) \to td_k$.

The algorithm takes the secret key SK_k from the *KeyGen* algorithm as the trapdoor td_k of the ABEKS scheme. Then we have

$$td_k = SK_k.$$

Test $(CT, PK, SK_k) \to \tilde{M}$.

Let $H(W)$ be the keyword set which encrypts the data file, and $(\mathbb{A}_k, \rho_k)$ be denoted as the access policy of user k. If $H(W)$ satisfies $\mathbb{A}_k$, then the algorithm finds a constant $\omega_{k,x}$, such that $\sum_{\rho_k(x) \in H(W)} \omega_{k,x} A_{k,x} = 1$. The algorithm computes

$$\prod_{\rho_k(x) \in H(W)} \frac{e(C_0, K_{k,x}^1)^{\omega_{k,x}}}{e(C_{\rho_k(x)}, K_{k,x}^2)^{\omega_{k,x}}}$$
$$= \prod_{\rho_k(x) \in H(W)} \frac{e(g,g)^{s\omega_{k,x}A_{k,x}u} e(g, X_1 T_{\rho_k(x)})^{s\omega_{k,x}r_{k,x}}}{e(g, X_1 T_{\rho_k(x)})^{s\omega_{k,x}r_{k,x}}}$$
$$= e(g,g)^{s \sum_{\rho_k(x) \in H(W)} \omega_{k,x}A_{k,x}u} = e(g,g)^{\alpha s}.$$

$\tilde{M} = C / e(g,g)^{\alpha s} = C \parallel 0^l$. Then we get the encrypted data file C.

Remark 1. We omit the description of the encryption algorithm of the data files because we can choose any appropriate encryption algorithm. For example, we can use a CP-ABE as the encryption algorithm, the attribute set can be the user's access account to the data servers or some private information distributed by the data servers. The employment of CP-ABE can prevent the collusion of the data server and the searchers. Since the data is encrypted by the attribute set, if a user is not authorized to get the data file, the cipher-text cannot be decrypted even if the user has searched the data successfully.

Remark 2. We use the access control matrix as the access policy in our ABEKS scheme. We have achieved a more accurate search policy by employing a more flexible access policy. This is just depending on the power of access policy.

Remark 3. Here is a restriction that each keyword can be used at most once in labeling the row of an LSSS matrix. However, we can relax the restriction by some useful tricks; for example, we can apply a encoding technique. We use the keywords $W : 1, W : 2, \ldots, W : k$ instead of keyword W. Then our keywords can be used k times.

4.4. The proof of security and consistency

In Section 4.2 we have proved that the underlying scheme is secure and attribute private, by Theorem 3.1 we can show that our scheme is computational consistent and keyword private.

5. Applications to cloud computing

Our scheme can also work as an efficient and secure searchable encryption scheme in cloud computing. In a cloud computing service, the data owner cannot predict the identities of users who want to execute a search on the uploaded data. Meanwhile, for the sake of data privacy, the data owner should be able to specify a group of users who are authorized to execute a search. We observed that ABE is perfectly suitable to this scenario. For example, we can adopt CP-ABE as the data file encryption algorithm, and use KP-ABE to construct our ABEKS scheme. Once the data owner has uploaded the data files to a remote server S, users can execute their own search on the data. The data owner firstly encrypts the data files $\{M_1, \ldots, M_n\}$ by using CP-ABE, which generates the ciphertext $\{C_1, \ldots, C_n\}$; then encrypts the CP-ABE ciphertext with the keyword set of the data file, which generates the ABEKS ciphertext $\{CT_1, \ldots, CT_n\}$. If a user U wants to search on the data, U should define a flexible access policy by the search policy, and get the secret key from the *KeyGen* algorithm. The secret key is sent to server S as trapdoor. S runs *Test* algorithm with the trapdoor and ABEKS ciphertext and returns the cipher-texts $\{CT_i\}$, which passes through the Test algorithm to U. At the same time, users B, C, D etc. can also execute their own search with their keywords. In this way we can achieve a flexible and secure search in a multi-user model.

6. Conclusion

In this paper we presented a general transformation from KP-ABE to ABEKS. By defining the data privacy and attribute privacy property of KP-ABE, we proved that an ABEKS scheme transformed from this KP-ABE was computational consistent and private. We proposed that the modified KP-ABE scheme could achieve this property, and we constructed a computational consistent ABEKS scheme based on this KP-ABE. In this ABEKS scheme, the searcher defined a more flexible search policy by constructing an access policy. Any user could execute a search on the data file shared in the storage server. It requires only that the user could generate a proper access policy from the keyword set.

Acknowledgments

We want to express our sincere thanks to the anonymous referees for their valuable comments and suggestions.

This work is supported by the National Nature Science Foundation of China under Grant Nos.: 60873041 and 61272091, the National Nature Science Foundation of Shandong Province under Grant Nos.: ZR2012FM005 and Y2007A13.

References

[1] Y.C. Chang, M. Mitzenmacher, Privacy preserving keyword searches on remote encrypted data, in: Applied Cryptography and Network Security, 2005, pp. 442–455.
[2] Eu-Jin Goh, Secure indexes, Cryptology ePrint Archive, Report 2003/216, 2003. http://eprint.iacr.org/2003/216/.
[3] D. Song, D. Wagner, A. Perrig, Practical techniques for searches on encrypted data, in: Proc. of the 2000 IEEE Symposium on Security and Privacy, 2000.
[4] W. Ogata, K. Kurosawa, Oblivious keyword search, Journal of Complexity 20 (2–3) (2004) 356–371. Special Issue on Coding and Cryptography.
[5] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohino, T. Lange, J.M. Lee, G. Neven, P. Pailliar, H. Shi, Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions, in: CRPYTO, 2005, pp. 205–222.
[6] D. Boneh, G. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, in: EUROCRYPT, 2005, pp. 440–456.
[7] A. Sahai, B. Waters, Fuzzy identity based encryption, in: EUROCRYPT, 2005, pp. 457–473.
[8] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute based encryption for fine-grained access control of encrypted data, in: ACM Conference on Computer and Communications Security, 2006, pp. 89–98.
[9] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute based encryption, in: IEEE Symposium on Security and Privacy, 2007, pp. 321–334.
[10] R. Ostrovsky, A. Sahai, B. Waters, Attribute based encryption with non-monotonic access structures, in: CCS, 2007.
[11] A. De Caro, V. Iovino, G. Persiano, Fully secure anonymous HIBE and secret-key anonymous IBE with short ciphertexts[M], in: Pairing-Based Cryptography—Pairing 2010, Springer, Berlin, Heidelberg, 2010, pp. 347–366.
[12] J.H. Seo, J.H. Cheon, Fully secure anonymous hierarchical identity-based encryption with constant size ciphertexts[J], Cryptology ePrint Archive, Report 2011/021, 2011. http://eprint.iacr.org.
[13] F. Han, J. Qin, H. Zhao, J. Hu, A general transformation from KP-ABE to searchable encryption, in: The 4th International Symposium on Cyberspace Safety and Security, CSS 2012.
[14] V. Goyal, A. Jain, O. Pandey, A. Sahai, Bounded ciphertext policy attributed based encryption, in: ICALP, 2008.
[15] M. Chase, Multi-authority attribute based encryption, in: TCC, 2007, pp. 515–534.
[16] A. Lewko, B. Waters, Decentralizing attribute based encryption, in: Eurocrypt 2011, in: LNCS, vol. 6632, 2011, pp. 568–588.
[17] B. Waters, Ciphertext-policy attribute based encryption: an expressive, efficient, and provable secure realization, in: PKC 2011, 2011, pp. 53–70.
[18] S. Hohenberger, B. Waters, Attribute-based encryption with fast decryption[M], in: Public-Key Cryptography—PKC 2013, Springer, Berlin, Heidelberg, 2013, pp. 162–179.
[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, in: LNCS, vol. 6110, 2010, pp. 62–91.
[20] B. Waters, Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions, in: CRYPTO, 2009, pp. 619–636.
[21] A. Lewko, B. Waters, New proof methods for attribute-based encryption: achieving full security through selective techniques, http://eprint.iacr.org/2012/326.pdf/.
[22] A. Beimel, Secure schemes for secret sharing and key distribution, Ph.D. Thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.
[23] D. Boneh, E. Goh, K. Nissim, Evaluating 2-DNF formulas on ciphertexts, in: TCC, 2005, pp. 325–342.

Fei Han is a Ph.D. candidate in School of Mathematics, Shandong University PR China. His current research interests focus on cryptographical protocols. Han is a member of the Chinese Association for Cryptologic Research (CACR) as well as the China Computer Federation (CCF).

Jing Qin is a professor in School of Mathematics, Shandong University PR China. Her research interests include computational number theory, information security, design and analysis of security about cryptologic protocols. She received her B.S. degree from Information Engineering University, Zhenzhou, PR China in 1982 and Ph.D. degree from School of Mathematics in Shandong University, PR China in 2004. She has co-authored 2 books and has published about 30 professional research papers. Prof. Qin is a senior member of the Chinese Association for Cryptologic Research (CACR) as well as the China Computer Federation (CCF).

Huawei Zhao is an associate professor of the school of Computer Science and Technology, Shandong University of Finance and Economics, Jinan. He received his M.S. degree in the School of Computer, and Ph.D. degree in the Institute of Information & Network Security from Shandong University in 2002 and 2006 respectively. He has published about 30 professional research papers. In 2011, he was a visiting scholar of the National Information Communication Technology of Australia (NICTA). His research interests include wireless network, network and information security.

Jiankun Hu is the Chair Professor and Research Director of Cyber Security Lab, School of Engineering and IT, University of New South Wales at the Australian Defence Force Academy (UNSW@ADFA), Canberra, Australia. He obtained his B.E. from Hunan University, China in 1983; Ph.D. in Control Engineering from Harbin Institute of Technology, China in 1993 and Masters by Research in Computer Science and Software Engineering from Monash University, Australia in 2000. He has worked in Ruhr University, Germany on the prestigious German Alexander von Humboldt Fellowship 1995–1996; as a research fellow in Delft University of the Netherlands 1997–1998, and a research fellow in Melbourne University, Australia 1998–1999. Jiankun's main research interest is in the field of cyber security, including biometrics security, where he has published many papers in high-quality conferences and journals including IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). He has served at the editorial board of up to 7 international journals and served as Security Symposium Chair of IEEE flagship conferences of IEEE ICC and IEEE Globecom. He has obtained 7 ARC (Australian Research Council) Grants and is now serving on the prestigious Panel of Mathematics, Information and Computing Sciences (MIC), ARC ERA (The Excellence in Research for Australia) Evaluation Committee.