Journal of Universal Computer Science, vol. 25, no. 3 (2019), 222-244
submitted: 28/3/18, accepted: 30/12/18, appeared: 28/3/19 © *J.UCS*

Towards Multi-user Searchable Encryption Supporting Boolean Query and Fast Decryption

Yunling Wang, Jianfeng Wang
(School of Cyber Engineering, Xidian University, Xi'an, P.R. China
ylwang0304@163.com, jfwang@xidian.edu.cn)

Shi-Feng Sun, Joseph K. Liu
(Faculty of Information Technology, Monash University, Melbourne, Australia
Shifeng.Sun@monash.edu, joseph.liu@monash.edu)

Willy Susilo, Joonsang Baek
(School of Computing and Information Technology
University of Wollongong, Wollongong, Australia
wsusilo@uow.edu.au, baek@uow.edu.au)

Ilsun You
(Department of Information Security Engineering
Soonchunhyang University, Asan, Republic of Korea
isyou@sch.ac.kr)

Xiaofeng Chen*
(School of Cyber Engineering, Xidian University, Xi'an, P.R. China
xfchen@xidian.edu.cn
*Corresponding author)

Abstract: Searchable encryption enables the data owner to outsource their data to the cloud server while retaining the search ability. Recently, some researchers proposed a variant of searchable encryption, named single-writer/multi-reader searchable encryption (SMSE), in which any authorized data user can perform a search query. That is, each document identifier is encrypted using attribute-based encryption (ABE), such that an arbitrary authorized user whose attributes match the corresponding access policy can access the document. However, the cloud server cannot determine whether the user has the ability to decrypt the matched data. Thus, it has to response all the search results to the data user, which causes a heavy communication and computation cost. To cope with this problem, we present a novel SMSE scheme based on server-side match technique, where the cloud can filter the documents that cannot be decrypted by the user and only return the matched ones. In addition, the decryption is also efficient, independent with the access policy structure. Security and efficiency evaluation show that our proposed scheme can achieve the desired security goals, while dramatically reducing the communication and computation overhead.

Key Words: Cloud computing, Searchable encryption, Multi-client, Fast decryption
Category: E.3

1 Introduction

Cloud computing provides on-demand computing resources over the internet on a pay-for-use basis. One of the most attractive benefits is for the resource-constrained users to outsource their data to the cloud server [Chen et al. 2015a, Chen et al. 2016, Li et al. 2019, Xue et al. 2019, Yu et al. 2017]. In this way, the users can enjoy the unlimited resources while reducing the maintenance cost locally. However, outsourcing data to the remote server brings some security challenges [Chen et al. 2014, Chen et al. 2015b], because the cloud server is curious about the users' data and tries to mining useful information. As a result, protecting the security of users' sensitive data is a critical demand in the outsourcing service. Using traditional encryption technique to encrypt data before outsourcing can protect the data security. However, it brings difficulties to perform some meaningful operations on the encrypted data, especially to search for some content interested [Yu et al. 2018A].

Fortunately, searchable encryption (SE) was proposed to protect the security of data while preserving the search ability over the encrypted data. Specifically, in SE scheme, the data owner first generates a search index and encrypts the documents, and then outsources them to the cloud server. Later, if the data owner wants to search some documents which contain a particular keyword, a search token is generated and submitted to the cloud server. Using the search token, the cloud server performs a search over the search index and finally returns the corresponding documents. For single keyword search, the search token is only for a single keyword such that all the final searched documents include the single queried keyword [Curtmola et al. 2006, Song 2000, Stefanov et al. 2014, Wang et al. 2015, Wang et al. 2017]. To rich the query expression, conjunctive searchable encryption was proposed [Ballard et al. 2005, Cash et al. 2013, Zuo et al. 2016], in which the final matched documents contain all the queried keywords. However, all these works focus on single-writer/single-reader searchable encryption (SSSE). That is, the data owner outsources the search index and documents, later it only allows the data owner itself to perform a search.

To extend SSSE to support data sharing, recently single-writer/multi-reader searchable encryption (SMSE) [Sun et al. 2016] was proposed. For SMSE, an authorized user is allowed to submit a legal search token and retrieve the corresponding documents. However, the challenge is how to allocate different permissions to the different authorized users. The permission contains two aspects, the search permission and the decryption permission. For the search permission, the data owner predefines a keyword set for each user, and an authorized user can generate a legal search token for the keywords in the predefines keyword set. For the decryption permission, an authorized user just can decrypt the search results which are delegated to him/her. This goal can be achieved by

encrypting the document identifiers based on attribute-based encryption (ABE) [Bethencourt et al. 2007]. However, it suffers from unnecessary communication overhead. The reason is that the server cannot determine whether the user can decrypt a particular ciphertext. Thus all the search results have to be returned to the user, including the ones which cannot be decrypted. This problem is severe when there are a large number of matched search results, but only a fraction can be decrypted. Besides, before performing decryption, the user has to test whether his/her attributes meet the decryption requirement of each ciphertext. As a result, it also brings the unnecessary computation overhead.

A trivial way to solve this problem is to send the user's attributes to the server. Then the server uses the attributes to match with the access policy and determine whether the user can decrypt the ciphertext. However, the attributes always contain the user's personal sensitive information, which should be protected well. Besides, the traditional ABE technique also leaks the access policy to the server. For the sake of protecting the access policy, anonymous ABE was proposed [Nishide et al. 2008]. However, it is very inefficient to perform decryption for the user. Recently, Zhang et al. [Zhang et al. 2017] proposed a match-then-decrypt technique to improve the decryption efficiency. The idea is for the user to perform match operation first to test whether s/he can decrypt the ciphertext. If yes, the user performs decryption. It indeed reduces a significant amount of computationally intensive operations, such as pairings. However, the match operation is performed on the user side. That means all the matched ciphertext still have to be transformed to the user including the ones which cannot be decrypted. Besides, the match operation also needs to consume the energy of the user's device. As far as the authors' knowledge, how to simultaneously reduce the unnecessary communication and decryption overhead in SMSE without leaking the access policy and user's attributes remains unsolved.

1.1 Our Contribution

In this paper, we proposed a new SMSE scheme, in which the server has an ability to test whether a ciphertext can be decrypted by the user. Thus the server just needs to transmit the ones which can be decrypted by the user. In this way, both the transmission and decryption cost are saved. Specifically, our contributions are two folds:

- We first proposed a server-side match technique for anonymous ABE. We design a particular component of ciphertext to test whether the access policy in it matches with the user's attributes. This match operation neither leaks the access policy in the ciphertext nor the user's attributes. Moreover, the decryption on the user side is very efficient. It just needs a constant number of pairing, which does not increase with the number of user's attributes.

- We then proposed a new SMSE scheme based on the server-side match technique. To achieve fine-grained access control, the document identifiers are encrypted with the proposed anonymous ABE. On receiving the search token, the server first finds out all the search results. Instead of immediately returning all of them to the user, the server uses the match technique to test whether they can be decrypted by the user. Finally, the server just returns the ones which can be decrypted by the user. In this way, both the communication and decryption overhead are dramatically reduced.

This is an extension of the conference version [Wang et al. 2017]. The main differences between this two versions are as follows: First, we add the related work in section 1.2, give the system model and design goals in section 3. Second, to make the scheme clearer, a high description of our proposed SMSE scheme is given in section 4.1. Third, we give more detailed proofs for the two lemmas in section 5.1. Finally, we provide thorough experimental results to evaluate the performance of our proposed scheme and compare it with the related work.

1.2 Related Work

Song et al. [Song 2000] proposed the first SE scheme, in which each keyword is encrypted based on a particular two-layer encryption structure. To perform a search query, it needs to scan the entire ciphertext sequentially. Goh [Goh 2003] improved the search efficiency by constructing an index for each document using Bloom Filter, which can quickly test whether an element in a set. However, the search complexity is linear in the number of documents. Besides, the Bloom Filter will bring false probability. The first sublinear scheme was proposed by Curtmola et al. [Curtmola et al. 2006]. In their scheme, the index is constructed based on the keyword set in the database (inverted index) other than each document. Subsequently, Kamara et al. [Kamara et al. 2012] extended Curtmola et al. scheme to support document update. Some other works [Stefanov et al. 2014, Kim et al. 2017, Bost et al. 2017, Song et al. 2018] also focus on dynamic property. However, all these works are single keyword search scheme.

More complex conjunctive keyword search was first proposed by Golle et al. [Golle et al. 2004]. Golle et al. gave two schemes for conjunctive keyword search. The trapdoor size in the first scheme is linear in the number of documents, while it is constant in the second one. However, the search index and search time are all linear in the number of documents. Some other works [Ballard et al. 2005, Ryu and Takagi 2007] are committed to improving the communication and storage efficiency. All these conjunctive keyword search schemes need a keyword field, which is not practical in some applications. Wang et al. [Wang et al. 2008] gave the first keyword field free conjunctive keyword search scheme. Later, Cash

et al. [Cash et al. 2013] proposed the first sublinear conjunctive keyword search using inverted index, which can also be extended to Boolean query. The idea is first to search the least frequent keyword and then match the search results with other queried keywords. Jarecki et al. [Jarecki et al. 2013] extended this to the SMSE setting, however, it requires an interaction between the user and the data owner for each query. Recently, Sun et al. [Sun et al. 2016] solved the interaction problem based on RSA assumption. In their scheme, the user can just decrypt the ciphertext which is delegated to him/her, because the identifiers are encrypted by ABE [Bethencourt et al. 2007].

ABE is a positive way to achieve fine-grained access control in data sharing system. Key-policy ABE (KP-ABE) [Goyal et al. 2006] and ciphertext-policy ABE (CP-ABE) [Bethencourt et al. 2007] are two kinds of ABE. They have different ways of encrypting a message. For KP-ABE, the ciphertext is related to a set of attributes and a user's private key is related to an access policy. While in CP-ABE, the ciphertext is associated with an access policy and a user's private key is associated with a set of attributes. Only when the user's attributes are satisfied with the access policy in the ciphertext, it can be decrypted by the user. CP-ABE is more practical in access control system because the data owner can encrypt a message under an access policy and share the ciphertext with different users. Since the first CP-ABE was proposed, plenty of works [Chase 2007, Yu et al. 2018, Li et al. 2014, Li et al. 2018, Yu et al. 2019] have been done to make improvement on it.

However, in these CP-ABE schemes, the access policy has to be sent along with the ciphertext, and a user needs to combine their attributes with the access policy to obtain the decryption key. To protect user's attribute privacy, anonymous ABE schemes [Kapadia et al. 2007, Lai et al. 2011] have been proposed. Although the attribute privacy is protected in these anonymous ABE schemes, they also suffer from a severe drawback of decryption efficiency. Zhang et al. [Zhang et al. 2017] proposed a match-then-decrypt anonymous ABE which improves the decryption efficiency. In this scheme, instead of directly decrypting a ciphertext, a match phase is performed on the user side to test whether the user can decrypt it.

1.3 Organization

The rest of this paper is organized as follows. Some necessary preliminaries are given in Section 2. The system architecture of our scheme is in Section 3. The proposed server-side match technique and the new SMSE scheme are described detailedly in Section 4. Next, we analyze the security of our scheme and compare it with the existing scheme in Section 5. Finally, the conclusions will be made in Section 6.

2 Preliminaries

2.1 Bilinear Pairings

Suppose $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic multiplicative groups with prime order p. Let g be a generator of $\mathbb{G}$. A bilinear pairing is a mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with the following properties:

1. Bilinear: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $g_1, g_2 \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$.
2. Non-degenerate: $e(g, g) \neq 1$.
3. Computable: It is efficient to compute $e(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}$.

2.2 Intractable Assumption

In this section, we first define some intractable assumptions in the cyclic multiplicative group $\mathbb{G}$ and then we define the strong RSA assumption.

Definition 1. The Decisional Diffie-Hellman (DDH) assumption holds in $\mathbb{G}$ if for any probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$ there exists a negligible function negl$(\cdot)$ such that

$$|\Pr[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1] - \Pr[\mathcal{A}(g, g^a, g^b, g^c) = 1]| = \text{negl}(n),$$

where g is randomly selected from $\mathbb{G}$, a, b, c are randomly selected from $\mathbb{Z}_p$ and n is the security parameter.

Definition 2. The Decisional Bilinear Diffie-Hellman (DBDH) assumption holds in $\mathbb{G}$ if for any PPT algorithm $\mathcal{A}$ there exists a negligible function negl$(\cdot)$ such that

$$|\Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 1] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, g^z) = 1]| = \text{negl}(n),$$

where g is randomly selected from $\mathbb{G}$, a, b, c, z are randomly selected from $\mathbb{Z}_p$ and n is the security parameter.

Definition 3. The Decision Linear (D-Linear) assumption holds in $\mathbb{G}$ if for any PPT algorithm $\mathcal{A}$ there exists a negligible function negl$(\cdot)$ such that
$|\Pr[\mathcal{A}(g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{z_3 + z_4} = 1] - \Pr[\mathcal{A}(g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^z) = 1]|$
$= \text{negl}(n)$,
where g is randomly selected from $\mathbb{G}$, z_1, z_2, z_3, z_4, z are randomly selected from $\mathbb{Z}_p$ and n is the security parameter.

Definition 4 Strong RSA assumption. Let p' and q' are primes and p and q are strong primes satisfied $p = 2p' + 1$ and $q = 2q' + 1$. The strong RSA assumption holds if any PPT algorithm $\mathcal{A}$ there exists a negligible function negl$(\cdot)$ such that

$$|\Pr[\mathcal{A}(n, g) = (z, e)]| = \text{negl}(\cdot),$$

which satisfies $z^e = g \mod n$, where $n = pq$ and g is a random element in $\mathbb{Z}_n^*$.

2.3 Anonymous ABE

Fine-grained access control in multi-user setting can be achieved by anonymous ABE. For each ciphertext, there is an access policy in it. For each user, s/he is allocated with a secret key according to her/his own attributes. If and only if the user's attributes satisfy the access policy, the user can decrypt the ciphertext with her/his secret key. In our proposed anonymous ABE, the server has an extra property that it can test whether a ciphertext can be decrypted by a user. This test will neither leak the access policy nor a user's attributes. The proposed anonymous ABE has four algorithms:

- **smABE.Setup**(κ): This setup algorithm takes as input the security parameter κ and outputs the public key pk and master key mk.
- **smABE.KeyGen**(L, pk, mk): This key generation algorithm takes as input the user's attribute set L, public key pk, and master key mk. It outputs the user's secret key $\mathbf{sk}_L$. The secret key includes two parts, $\mathbf{sk}_{mat}$ and $\mathbf{sk}_{dec}$. $\mathbf{sk}_{mat}$ is used for matching and $\mathbf{sk}_{dec}$ is used for decrypting.
- **smABE.Enc**(M, m, P): This encryption algorithm takes as input a message M, an auxiliary information m for M and access policy P. It outputs the ciphertext $\mathbf{e}$ for M. In our scheme, every message M is associated with an information m. The ciphertext $\mathbf{e}$ can be decrypted by the user whose attribute set L satisfies the access policy P.
- **smABE.Match**$(\mathbf{e}, mt, D_{\Delta,0})$: This match algorithm is to test whether a particular user can decrypt the ciphertext $\mathbf{e}$. It inputs the ciphertext $\mathbf{e}$, auxiliary information mt and $D_{\Delta,0}$ associated with user's secret key. If the ciphertext matches with the user's attribute set, it outputs the ciphertext $\mathbf{e}_{dec}$ for decrypting. Otherwise, it outputs "no".
- **smABE.Dec**$(\mathbf{e}_{dec}, \mathbf{sk}_{dec})$: This decryption algorithm takes as input the ciphertext $\mathbf{e}_{dec}$ and user's decryption key $\mathbf{sk}_{dec}$. It outputs the corresponding message M.

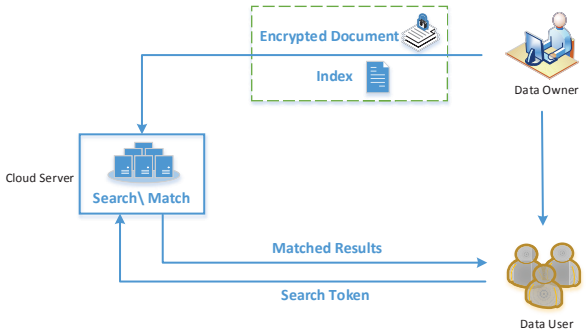


Figure 1: System Model

3 System Architecture

3.1 System Model

Our SMSE scheme supporting fast decryption consists of three entities: the data owner, the cloud server and the user. The data owner has a plaintext database and wants to outsource it to the cloud server. The data owner first encrypts the database and generates a search index, then outsources them to the server. Besides, the data owner allocates different privileges for searching and decrypting to every authorized user. When a user wants to search for some documents, she should generate a valid search token and submit it to the server. Upon receiving the search token, the server performs a search over the search index and finds out the corresponding search results. After that, the server tests each search result to determine whether the user can decrypt. Finally, the server just returns the search results which can be decrypted. In our system, we assume the server is "honest-but-curious", which means the server will follow our protocol and return the correct search results, but try to learn as much information as possible from the protocol. Figure 1 shows the system model. Formally, our scheme consists of five algorithms:

- **SMSE.Setup**$(\lambda, \mathrm{DB}, \mathrm{K}, \mathcal{U})$: Takes as input the security parameter λ, the plaintext database DB, a list of decryption key array K and universe attribute set $\mathcal{U}$, this setup algorithm outputs the system master key MK, public key PK, the encrypted search index TSet and XSet. Finally the TSet and XSet are outsourced to the server.
- **SMSE.KGen**$(\mathrm{MK}, L, \mathbf{w})$: Takes as input the system master key MK, user's attribute set L and a query keyword set $\mathbf{w}$, this key generation algorithm outputs the user's private key sk. Finally, sk and $\mathbf{w}$ are together sent to the

user. The user can just generate the search tokens for the keywords in the set $\mathbf{w}$.

- **SMSE.TGen**(sk, Q): Takes as input the private key sk and query Q, this token generation algorithm outputs the search token st for the query Q.
- **SMSE.Search**$(st, \text{TSet}, \text{XSet})$: Takes as input the search token st, the search index TSet and XSet, the server firstly searches over the index TSet and XSet, finds out the corresponding search results R, then determines whether the user can decrypt the results, finally returns the ones in S that the user can decrypt.
- **SMSE.Retrieve**(sk, R): Takes as input the private key sk and the final search results in S from the server, the user first decrypts the ciphertexts in S and gets the document identifiers and document keys k_{id}, then retrieves the corresponding documents and decrypts them by k_{id}.

3.2 Design Goals

This work aims to achieve multi-user searchable encryption supporting fast decryption. Based on the "honest-but-curious" server model, the design goals of our scheme are as follows:

- **Data Confidentiality**. It is required that an adversary cannot learn the data owner's original data from the outsourced data stored on the cloud server. In addition, the adversary cannot learn the queried keywords from the search token.
- **Attribute Privacy**. The cloud server can neither learn the access policy in the ciphertext, nor the user's attributes.
- **Fast Decryption**. The cloud server can test whether a user can decrypt a particular ciphertext with the help of the search token. But this test should only work among the search results for the search token rather than the whole ciphertext on the cloud server. After receiving the matched searched results, the user should decrypt them efficiently.

3.3 Security Definition

The security of our proposed scheme is defined by a leakage function $\mathcal{L}$, which is learned by an adversary during the interaction with the proposed scheme. The security demonstrates that the proposed scheme leaks nothing rather than the leakage function. Specifically, an adversary has an negligible probability to distinguish whether it is in a real experiment $\mathbf{Real}_{\mathcal{A}}^{\Pi}(\lambda)$ or in an ideal experiment

$\mathbf{Ideal}^{\Pi}_{\mathcal{A},\mathcal{S}}(\lambda)$. The ideal experiment is simulated by a simulator $\mathcal{S}$ based on the leakage function. We describe the two experiments as follows:

$\mathbf{Real}^{\Pi}_{\mathcal{A}}(\lambda)$: The adversary $\mathcal{A}$ chooses a database DB, and then the experiment runs the setup algorithm $(\mathrm{PK}, \mathrm{MK}, \mathrm{TSet}, \mathrm{XSet}) \leftarrow \mathbf{SMSE.Setup}(\lambda, \mathrm{DB}, \mathrm{K}, \mathcal{U})$, and the adversary is given $(\mathrm{PK}, \mathrm{TSet}, \mathrm{XSet})$. Later, $\mathcal{A}$ selects an authorized keyword set $\mathbf{w}$ and gets the corresponding secret key. Next, $\mathcal{A}$ repeatedly and adaptively chooses a query Q, and the experiment runs the token generation and search algorithms. Following, the experiment gives $\mathcal{A}$ the transcript (the search token and the search results for the query) and the corresponding document identifiers. Finally, $\mathcal{A}$ returns a bit as the experiment output.

$\mathbf{Ideal}^{\Pi}_{\mathcal{A},\mathcal{S}}(\lambda)$: The adversary $\mathcal{A}$ chooses a database DB. Then the experiment gives the $(\mathrm{PK}, \mathrm{TSet}, \mathrm{XSet})$ to the adversary using the simulator $\mathcal{S}$, $(\mathrm{PK}, \mathrm{TSet}, \mathrm{XSet}) \leftarrow \mathcal{S}(\mathcal{L}(\mathrm{DB}))$. Later, $\mathcal{A}$ repeatedly and adaptively chooses a query Q. With the query, the experiment outputs the transcript and the corresponding document identifiers by $\mathcal{S}(\mathcal{L}(\mathrm{DB}, \mathbf{Q})$, where $\mathbf{Q}$ represents all the previous queries and the latest query. Finally, $\mathcal{A}$ returns a bit as the experiment output.

Definition 5. Our proposed scheme Π is $\mathcal{L}$-semantically secure against adaptive attacks if for all PPT adversaries $\mathcal{A}$ there is a simulator $\mathcal{S}$ such that $|\Pr[\mathbf{Real}^{\Pi}_{\mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Ideal}^{\Pi}_{\mathcal{A},\mathcal{S}}(\lambda) = 1]| \leq negl(\lambda)$.

4 SMSE Scheme with Fast Decryption

4.1 High Description

In this paper, we propose a new SMSE scheme supporting Boolean queries and fast decryption, which saves both the communication and decryption overhead. First, we propose a server-side match technique for anonymous ABE. This technique enables the server to test whether a particular ciphertext can be decrypted by a user. To achieve this goal, we introduce special components in the ciphertext $(C_{\Delta}, C_x, \hat{C}_0, C_{i,t,\Delta})$. Under the help of mt and $D_{\Delta,0}$ which contain the information of the user's attributes, the server can determine whether the user's attributes meet the access policy hidden in the ciphertext. Note that this match technique neither leaks the access policy nor the user's attributes. Second, we use our new anonymous ABE to encrypt the document identifiers instead of the traditional ABE scheme. In this way, the server can use the match technique to return the ones which can be decrypted by the user, rather than the entire search results. As a result, both the communication and decryption overhead are reduced.

4.2 Server-side Match Technique for Anonymous ABE

In this section, we present our server-side match technique for anonymous ABE (smABE).

- **smABE.Setup**(κ): We denote two cyclic multiplicative groups of prime order p as $\mathbb{G}$ and $\mathbb{G}_T$, and a bilinear map e as $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We also assume the system attribute set is $\mathcal{U} = \{\tau_1, \tau_2, ..., \tau_n\}$ and each attribute has multiple values, where $\tau_i = \{\nu_{i,1}, \nu_{i,2}, ..., \nu_{i,n_i}\}$. The data owner randomly chooses $g_1, g_2 \xleftarrow{R} \mathbb{G}$ and $y \xleftarrow{R} \mathbb{Z}_p$, then computes $Y \leftarrow e(g_1, g_2)^y$. The system public key is $pk = \langle g_1, g_2, Y \rangle$ and the system master key is $mk = \langle y \rangle$.

- **smABE.KeyGen**(L, pk, mk): This algorithm generates the attribute secrete key $\mathbf{sk}_L$ for a certain user whose attribute set is $L = \{L_1, L_2, ..., L_n\}$. H is a hash function: $\{0,1\}^* \to \mathbb{G}$. First, the data owner randomly chooses $r, \lambda, \hat{\lambda}$ from $\mathbb{Z}_p$ and computes $D_{\Delta,0} \leftarrow g_1^r$, $D_x \leftarrow g_2^r$, $D_0 \leftarrow g_2^{\lambda}$, $\hat{D}_0 \leftarrow g_1^{\hat{\lambda}}$. Assume $L_i = \nu_{i,k_i}$, the data owner also computes $\hat{D}_{\Delta,0} \leftarrow g_2^y \prod_{i=1}^n H(i||\nu_{i,k_i})^r$, $D_1 \leftarrow g_1^y \prod_{i=1}^n H(0||i||\nu_{i,k_i})^{\lambda}$ and $\hat{D}_1 \leftarrow g_2^y \prod_{i=1}^n H(1||i||\nu_{i,k_i})^{\hat{\lambda}}$. Then the attribute secrete key is $\mathbf{sk}_L = \langle \mathbf{sk}_{mat}, \mathbf{sk}_{dec} \rangle$, where $\mathbf{sk}_{mat} = \langle D_{\Delta,0}, \hat{D}_{\Delta,0}, D_x \rangle$ and $\mathbf{sk}_{dec} = \{D_0, \hat{D}_0, D_1, \hat{D}_1\}$.

- **smABE.Enc**(M, m, P): The data owner encrypts a message $M \in \mathbb{G}_T$ under the policy of $P = \{P_1, P_2, ..., P_n\}$. In our scheme, every massage M is related to an auxiliary information $m \in \mathbb{Z}_p$. The data owner first generates $s, s', s'' \xleftarrow{R} \mathbb{Z}_p$, and then computes $\tilde{C} \leftarrow MY^s$; $C_{\Delta} \leftarrow Y^{s'}$; $\hat{C}_0 \leftarrow g_1^{s'}$; $C_1 \leftarrow g_2^{s''}$; $\hat{C}_1 \leftarrow g_1^{s-s''}$; $C_x \leftarrow g_2^{s'm}$. Then the data owner chooses $\{\sigma_{i,\Delta}, \sigma_{i,0}, \sigma_{i,1} \xleftarrow{R} \mathbb{G} | 1 \le i \le n\}$ such that $\prod_{i=1}^n \sigma_{i,\Delta} = \prod_{i=1}^n \sigma_{i,0} = \prod_{i=1}^n \sigma_{i,1} = 1_{\mathbb{G}}$ and computes $[C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}]$ as follows:

  1. If $v_{i,t} \notin P_i$, $[C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}] \xleftarrow{R} \mathbb{G}$.
  2. If $v_{i,t} \in P_i$, $[C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}] \leftarrow$
  $[\sigma_{i,\Delta} H(i||\nu_{i,t})^{s'}, \sigma_{i,0} H(0||i||\nu_{i,t})^{s''}, \sigma_{i,1} H(1||i||\nu_{i,t})^{s-s''}]$.

  Finally, the ciphertext of M is

$$\mathbf{e} = \langle C_{\Delta}, C_x, \hat{C}_0, \tilde{C}, C_1, \hat{C}_1, \{\{C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}\}_{1 \le t \le n_i}\}_{1 \le i \le n} \rangle.$$

- **smABE.Match**($\mathbf{e}, mt, D_{\Delta,0}$): The server performs this algorithm to test whether a certain user can decrypt the ciphertext $\mathbf{e}$. Here mt and $D_{\Delta,0}$ are given by the user, where $mt = \hat{D}_{\Delta,0} \cdot D_x^m$. This algorithm outputs "yes" if

$$e(D_{\Delta,0}, C_x \prod_{i=1}^{n} C_{i,t,\Delta}) C_{\Delta} = e(\hat{C}_0, mt).$$

  According to $C_{i,t,\Delta}$, the server finds all the corresponding $C_{i,t,0}$ and $\hat{C}_{i,t,0}$, then computes $C \leftarrow \prod_{i=1}^n C_{i,t,0}$, $\hat{C} \leftarrow \prod_{i=1}^n \hat{C}_{i,t,0}$. Finally, the server sends $\mathbf{e}_{dec} = \langle \tilde{C}, C_1, \hat{C}_1, C, \hat{C} \rangle$ to the user.

- **smABE.Dec**($\mathbf{e}_{dec}, \mathbf{sk}_{dec}$): The user performs this algorithm to decrypt the ciphertext and gets the M.

$$M \leftarrow \frac{\tilde{C}e(C, D_0)e(\hat{D}_0, \hat{C})}{e(D_1, C_1)e(\hat{C}_1, \hat{D}_1)}.$$

4.3 Our Construction

In this section, we present our SMSE scheme with fast decryption. We denote the whole keyword set as $\mathrm{W} = \bigcup_{i=1}^{d} \mathrm{W}_{id_i}$, where W_{id_i} represents the keyword set for document d_i; the data owner's database as $\mathrm{DB} = (id_i, \mathrm{W}_{id_i})_{i=1}^{d}$; the identifiers set containing keyword w as $\mathrm{DB}[w] = \{id : w \in \mathrm{W}_{id}\}$; the decryption key set as K which is used to decrypt the original documents. And we let λ be the security parameter. The details of the proposed scheme are given as follows.

- **SMSE.Setup**($\lambda, \mathrm{DB}, \mathrm{K}, \mathcal{U}$): This algorithm is run by the data owner. Firstly, it chooses two big primes p, q, and computes $N = pq$. It denotes two pseudo-random functions $F : \{0,1\}^{\lambda} \times \{0,1\}^{\lambda} \rightarrow \{0,1\}^{\lambda}$, $F_p : \{0,1\}^{\lambda} \times \{0,1\}^{\lambda} \rightarrow \mathbb{Z}_p^*$ and selects random keys K_I, K_Z, K_X for the F_p and K_S for the F. Let H: $\{0,1\}^* \rightarrow \mathbb{Z}_p$ be a hash function. The data owner randomly chooses $g \xleftarrow{R} \mathbb{G}$ and $\tilde{g}_1, \tilde{g}_2, \tilde{g}_3 \xleftarrow{R} \mathbb{Z}_N^*$. Then it outputs the system master key $\mathrm{MK} = \langle p, q, K_S, K_I, K_Z, K_X, \tilde{g}_1, \tilde{g}_2, \tilde{g}_3 \rangle$ and system public key $\mathrm{PK} = \langle N, g \rangle$. The search index TSet and XSet are generated as in Algorithm 1 and finally they are sent to the server.

- **SMSE.KGen**($\mathrm{MK}, L, \mathbf{w}$): This algorithm is run by the data owner. Suppose that an authorized user with attribute set L can search over keyword set $\mathbf{w} = \{w_1, w_2, ..., w_n\}$, where the appearance frequency of the keywords satisfies $|w_1| < |w_2| < \cdots < |w_n|$. The attribute private key is computed as $sk_L = \mathbf{smABE.KeyGen}(L, pk, mk) = \langle \mathbf{sk}_{mat}, \mathbf{sk}_{dec} \rangle$ for a user whose attribute set is $L = \{L_1, L_2, ..., L_n\}$. For search keyword set $\mathbf{w}$, the search private key $sk_{\mathbf{w}} = \{sk_{\mathbf{w}}^{(1)}, sk_{\mathbf{w}}^{(2)}, sk_{\mathbf{w}}^{(3)}\}$ is computed as:

$$sk_{\mathbf{w}}^{i} = (\tilde{g}_i^{1/\prod_{j=1}^{n} w_j} \mod N), \quad i = \{1, 2, 3\}.$$

  Finally, the user's private key $sk = \{K_S, K_Z, K_X, sk_{\mathbf{w}}, sk_L\}$ and $\mathbf{w}$ are all sent to the user.

- **SMSE.TGen**(sk, Q): This algorithm is run by the authorized user. Suppose that an authorized user wants to perform conjunctive keyword search for $\bar{\mathbf{w}} = \{w_1', w_2', \cdots, w_m'\}$, where $m \leq n$ and w_1' is the least frequency keyword among the queried keywords. Then the search token st is generated as follows and finally it is sent to the server.

Algorithm 1 Search Index (TSet, XSet) Generated Algorithm

Input: MK, PK, DB, K
Output: TSet, XSet

TSet, XSet $\leftarrow \phi$
for $w \in$ W **do**
  $stag_w \leftarrow F(K_S, \tilde{g}_1^{1/w} \mod N)$; $m \leftarrow H(\tilde{g}_1^{1/w} \mod N)$; $c \leftarrow 1$
  for $id \in$ DB$[w]$ **do**
    $xind \leftarrow F_p(K_I, id)$; $z \leftarrow F_p(K_Z, \tilde{g}_2^{1/w} \mod N||c)$; $l \leftarrow F(stag_w, c)$
    $y \leftarrow xind \cdot z^{-1}$;
    $\mathbf{e} \leftarrow$ **smABE.Enc**$((id||k_{id}), m, P)$
    TSet$[l] = (\mathbf{e}, y)$
    $xtag_w \leftarrow g^{F_p(K_X, \tilde{g}_3^{1/w} \mod N) \cdot xind}$; XSet $\leftarrow$ XSet $\cup \{xtag_w\}$
    $c \leftarrow c + 1$
  end for
end for
return TSet, XSet

- $stag \leftarrow F(K_S, (sk_{\mathbf{w}}^{(1)})^{\prod_{w \in \mathbf{w} \setminus w_1'} w} \mod N) = F(K_S, \tilde{g}_1^{1/w_1'} \mod N)$
- For $c = 1, 2, \cdots$ until the server say stop
  1. For $i = 2, ..., m$

  $xt[c, i] \leftarrow g^{F_p(K_Z, (sk_{\mathbf{w}}^{(2)})^{\prod_{w \in \mathbf{w} \setminus w_1'} w} \mod N||c) \cdot F_p(K_X, (sk_{\mathbf{w}}^{(3)})^{\prod_{w \in \mathbf{w} \setminus w_i'} w} \mod N)}$
  $= g^{F_p(K_Z, \tilde{g}_2^{1/w_1'} \mod N||c) \cdot F_p(K_X, \tilde{g}_3^{1/w_i'} \mod N)}$;

  2. Set $xt[c] = xt[c, 2], \cdots, xt[c, m]$;
- $mt \leftarrow \hat{D}_{\Delta,0} \cdot D_x^{H(\tilde{g}_1^{1/w_1'} \mod N)}$;
- set $st = (mt, D_{\Delta,0}, stag, xt[1], xt[2], \cdots)$.

- **SMSE.Search**$(st, \text{TSet}, \text{XSet})$. This algorithm is run by the server which consists of two steps: *Search Step* and *Match Step*. The details are shown in Algorithm 2.
  - *Search Step*: The server first finds out all the encrypted identifiers which are satisfied with the search token. Specifically, The server uses *stag* to find out the encrypted identifiers containing w_1', and then justifies whether they contain the other queried keywords $(w_2', \cdots, w_m')$. Finally, this step outputs the encrypted identifiers which contain all the queried keywords.

- *Match Step*: The server then uses mt and $D_{\Delta,0}$ to test whether the user can decrypt the encrypted identifiers generated in the *Search Step*. Finally, this *Match Step* outputs the ultimate encrypted identifiers which can be decrypted by the user.

- **SMSE.Retrieve**(sk, R). This algorithm is run by the user. On input the private key sk and the search result R, this algorithm outputs the identifiers and the corresponding document keys. Finally the user retrieves the documents and decrypts them.

  - For each $\hat{\mathbf{e}} \in R$, parse $\hat{\mathbf{e}} = \langle \mathbf{e}_{dec} \rangle$ and the user directly decrypts the ciphertext

$$(id \parallel k_{id}) \leftarrow \mathbf{smABE.Dec}(\mathbf{e}_{dec}, \mathbf{sk}_{dec}).$$

  - For each id, the user sends it to the server and then gets the corresponding encrypted document. Finally the encrypted document can be decrypted by k_{id}.

Algorithm 2 SMSE.Search$(st, \text{TSet}, \text{XSet})$

Input: $st, \text{TSet}, \text{XSet}$
Output: R

$S, R \leftarrow \phi$
Parse $st = (mt, D_{\Delta,0}, stag, xt[1], xt[2], \cdots)$
$c = 1$; $l \leftarrow F(stag, c)$
while TSet$[l]$ exist **do**
    $(\mathbf{e}, y) \leftarrow \text{TSet}[l]$
    if $xt[c, i]^y \in$ XSet for all i **then**
        $S \leftarrow S \cup \{e\}$
    end if
    $c \leftarrow c + 1$; $l \leftarrow F(stag, c)$
end while
for $\mathbf{e} \in S$ **do**
    if **smABE.Match**$(\mathbf{e}, mt, D_{\Delta,0}) =$ "yes" **then**
        set $\hat{\mathbf{e}} \leftarrow \langle \mathbf{e}_{dec} \rangle$; $R \leftarrow R \cup \{\hat{\mathbf{e}}\}$
    end if
end for
return R

5 Analysis of Our Proposed Scheme

5.1 Security Analysis

We first describe the security model for our smABE scheme using the following game. A scheme is defined as IND-sCP-CPA security if no PPT adversary can break this game with a non-negligible advantage.

Init: The adversary $\mathcal{A}$ submits two challenge access policies P_0 and P_1.
Setup: The challenger $\mathcal{C}$ runs the **smABE.Setup** algorithm and gives the public key PK to the $\mathcal{A}$.
Phase 1: $\mathcal{A}$ submits an attribute list L to the $\mathcal{C}$. $\mathcal{C}$ runs **smABE.KeyGen** and returns the secret key $\mathbf{sk}_L$, if $(L \models P_0 \wedge L \models P_1)$ or $(L \not\models P_0 \wedge L \not\models P_1)$. $\mathcal{A}$ can repeat this quary polynomial times.
Challenge: $\mathcal{A}$ submits two messages M_0 and M_1 to the $\mathcal{C}$. If any attribute list satisfies both P_0 and P_1, it is required that $M_0 = M_1$. $\mathcal{C}$ randomly chooses a bit $v \in 0, 1$, computes $\mathbf{e}_{P_v} = \mathbf{smABE.Enc}(M_v, m, P_v)$ and sends $\mathbf{e}_{P_v}$ to $\mathcal{A}$.
Phase 2: Repeat the **Phase 1**. $\mathcal{A}$ cannot submit L which satisfies $L \models P_0 \wedge L \models P_1$, if $M_0 \neq M_1$.
Guess: $\mathcal{A}$ outs a guess v' of v. The advantage of $\mathcal{A}$ in this game is defined as

$$\mathbf{Adv}_{\mathrm{smABE}}^{\mathrm{IND-sCP-ABE}} = |\Pr[v' = v] - \tfrac{1}{2}|$$

Fig.1. The IND-sCP-CPA game

Theorem 6. *Our smABE scheme is IND-sCP-CPA secure under the DBDH assumption and D-linear assumption. The advantage* ϵ_{CPA} *for a PPT adversary to attack the IND-sCP-CPA game in the random oracle model is negligible.*

Proof. A sequence of hybrid games are used to prove that $\mathcal{A}$ cannot win the original game G with non-negligible probability. It is supposed that the two challenge access policies $P_0 = [P_{0,1}, P_{0,2}, \cdots, P_{0,n}]$ and $P_1 = [P_{1,1}, P_{1,2}, \cdots, P_{1,n}]$ are submitted at the beginning of the game. We first modify game G to game G_0. In game G_0, if $\mathcal{A}$ obtains the secret attribute key sk_L when $(L \not\models P_0 \wedge L \not\models P_1)$, the ciphertext component $\tilde{C}$ is randomly chosen in G_T and the rest components are generated as usual. If $\mathcal{A}$ obtains the secret attribute key sk_L when $(L \models P_0 \wedge L \models P_1)$, all components for the ciphertext are generated like in game G. In this case, $G_0 = G_1$. Then we change the components $\{\{C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}$ and define a sequence of other games as follows.

For every attribute value $\nu_{i,t}$ in the universe attribute set, if $(\nu_{i,t} \in P_{0,i} \wedge \nu_{i,t} \in P_{1,i})$ or $(\nu_{i,t} \notin P_{0,i} \wedge \nu_{i,t} \notin P_{1,i})$, the components $\{C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}\}$ in all games are generated in normal way like in game G. If there exists a $\nu_{i,t}$ such that $(\nu_{i,t} \in P_{0,i} \wedge \nu_{i,t} \notin P_{1,i})$ or $(\nu_{i,t} \notin P_{0,i} \wedge \nu_{i,t} \in P_{1,i})$, the ciphertext components $\{C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}\}$ generated normally in game G_{l-1} will be replaced in game G_l by a random element in group $\mathbb{G}$. We stop this replace process when there is no such $\nu_{i,t}$ satisfies $(\nu_{i,t} \in P_{0,i} \wedge \nu_{i,t} \notin P_{1,i})$ or $(\nu_{i,t} \notin P_{0,i} \wedge \nu_{i,t} \in P_{1,i})$.

In the last game, the advantage of $\mathcal{A}$ is zero because the distribution of the ciphertext components are the same no matter what the random bit ν is. The above games are denoted as $\{G, G_0, G_1, \cdots, G_{l_{max}}\}$. Then we denote the probability for $\mathcal{A}$ to win the original game G as $\Pr[\varepsilon]$ and the probability to win the game G_l is denoted as $\Pr[\varepsilon_l]$. Then the advantage in game G_0 can be represented as ϵ_{CPA}, where $\epsilon_{\text{CPA}} = |\Pr[\varepsilon] - \frac{1}{2}| = |\Pr[\varepsilon] - \Pr[\varepsilon_{l_{max}}]|$, which has the following property.

$$\epsilon_{\text{CPA}} \leq |\Pr[\varepsilon] - \Pr[\varepsilon_0]| + \sum_{l=1}^{l_{max}} |\Pr[\varepsilon_{l-1}] - \Pr[\varepsilon_l]|.$$

We can prove that ϵ_{CPA} is negligible under the assumption of DBDH and D-linear. From the Lemma 1 and Lemma 2, the inequalities $|\Pr[\varepsilon] - \Pr[\varepsilon_0]| \leq \epsilon_{\text{DBDH}}$ and $|\Pr[\varepsilon_{l-1}] - \Pr[\varepsilon_l]| \leq \epsilon_{\text{DL}}$ are hold. Here the ϵ_{DBDH} and ϵ_{DL} represent the advantage for a distinguisher to win the DBDH challenge and D-linear challenge. Thus the inequality $\epsilon_{\text{CPA}} \leq \epsilon_{\text{DBDH}} + |\mathcal{U}|\epsilon_{\text{DL}}$ holds, where $|\mathcal{U}|$ represents the number of attributes in the system. Under the DBDH and D-linear assumptions, ϵ_{DBDH} and ϵ_{DL} are negligible and thus ϵ_{CPA} is negligible. So we conclude that our proposed scheme smABE is IND-sCP-CPA secure under the DBDH assumption and D-linear assumption.

Lemma 7. *The probability difference for a PPT adversary $\mathcal{A}$ to win the game G and game G_0 is negligible under the DBDH assumption, that is* $|\Pr[\varepsilon] - \Pr[\varepsilon_0]| \leq \epsilon_{\text{DBDH}}$.

Proof. We build a distinguisher $\mathcal{D}$ to play the DBDH game. Given a challenge of $[g, g^a, g^b, g^c, Z]$, $\mathcal{D}$ distinguishes whether Z is $e(g,g)^{abc}$ or a random element in $\mathbb{G}_T$. The details are as follows:

For *Init*, A commites two challenge access policy $P_0^* = [P_{0,1}^*, P_{0,2}^*, \cdots, P_{0,n}^*]$ and $P_1^* = [P_{1,1}^*, P_{1,2}^*, \cdots, P_{1,n}^*]$ to the $\mathcal{D}$. Then $\mathcal{D}$ randomly selects a bit $v \in 0, 1$. Next, for *Setup*, $\mathcal{D}$ randomly chooses $\omega \xleftarrow{R} \mathbb{Z}_p$, sets $g_1 = g^\omega$, $g_2 = g^b$ and $Y = e(g_1, g_2)^a = e(g_1, g_2)^y$. Then the system public key is $PK = \langle g, g_1, g_2, Y \rangle$ and sends PK to $\mathcal{A}$. For *Phase 1*, $\mathcal{A}$ visit the hash oracle and key generation oracle to get the secret key $\mathbf{sk_L}$ for attribute set $\mathbf{L}$, where $(L \nvDash P_0^* \wedge L \nvDash P_1^*)$. For *Challenge*, $\mathbb{A}$ submits two challenge ciphertext M_0 and M_1 to $\mathcal{D}$. Then $\mathcal{D}$ sets $\tilde{C} = M_v Z^\omega$, and the rest ciphertext components are generated in normal way. For *Phase 2*, the distinguisher does as in *Phase 1*. For *Guess*, $\mathcal{A}$ output a bit v'. If $v = v'$, $\mathcal{D}$ outputs 1, otherwise $\mathcal{D}$ outputs 0. In case 1, $Z = e(g,g)^{abc}$ and $\tilde{C} = M_v Z^\omega = M_v Y^c$. The view of $\mathcal{A}$ in this case is distributed exactly as $\mathcal{A}'s$ view in G. We have $\Pr[\mathcal{D}(g, g^a, g^b, g^c, e(g,g)^{abc})] = \Pr[\varepsilon]$. In case 2, Z is a random element in $\mathbb{G}_T$ where $Z = g^z$. The view of $\mathcal{A}$ in this case is distributed exactly as $\mathcal{A}'s$ view in G_0. We have $\Pr[\mathcal{D}(g, g^a, g^b, g^c, g^z)] = \Pr[\varepsilon_0]$. So $|\Pr[\varepsilon] - \Pr[\varepsilon_0]| = |\Pr[\mathcal{D}(g, g^a, g^b, g^c, e(g,g)^{abc})] - \Pr[\mathcal{D}(g, g^a, g^b, g^c, g^z)]| \leq \epsilon_{DBDH}$.

Lemma 8. *The probability difference for a PPT adversary $\mathcal{A}$ to win the game G_{l-1} and game G_l is negligible under the D-Linear assumption, that is* $|\Pr[\varepsilon_{l-1}] - \Pr[\varepsilon_l]| \leq \epsilon_{\mathrm{DL}}$ *for* $1 \leq l \leq l_{max}$.

Proof. We build a distinguisher $\mathcal{D}_l$ to play the D-Linear game. Given a challenge of $[g, g^{z_1}, g^{z_2}, Z, g^{z_2 z_4}, g^{z_3+z_4}]$, $\mathcal{D}_l$ distinguishes whether Z is $g^{z_1 z_3}$ or a random element in $\mathbb{G}_T$. From Theorem 1, the ciphertext components $\{C_{i_l,t_l,\Delta}, C_{i_l,t_l,0}, \hat{C}_{i_l,t_l,0}\}$ generated normally in $\mathbf{G}_{l-1}$ are replaced by a random element from $\mathbb{G}$ in game $\mathbf{G}_l$. In general, suppose that $(\nu_{i_l,t_l} \notin P^*_{0,i_l} \wedge \nu_{i_l,t_l} \in P^*_{1,i_l})$. The details are as follows:

For *Init*, A commits two challenge access policy $P^*_0 = [P^*_{0,1}, P^*_{0,2}, \cdots, P^*_{0,n}]$ and $P^*_1 = [P^*_{1,1}, P^*_{1,2}, \cdots, P^*_{1,n}]$ to the $\mathcal{D}_l$. Then $\mathcal{D}_l$ randomly selects a bit $v \in 0, 1$. Next, for *Setup*, $\mathcal{D}_l$ sets $g_1 = g^{z_1}$, $g_2 = g^{z_2}$, randomly selects $y \xleftarrow{R} \mathbb{Z}_p$ and sets $Y = e(g_1, g_1)^y$. Then the system public key is $PK = \langle g, g_1, g_2, Y \rangle$ and sends PK to $\mathcal{A}$. For *Phase 1*, $\mathcal{A}$ visit the hash oracle and key generation oracle to get the secret key $\mathbf{sk_L}$ for attribute set $\mathbf{L}$. For *Challenge*, $\mathbb{A}$ submits two challenge ciphertext M_0 and M_1 to $\mathcal{D}$. If $(L \not\models P^*_0 \wedge L \not\models P^*_1)$, $\tilde{C}$ is a random element in $\mathbb{G}_T$. Otherwise, $\mathcal{D}_l$ sets $\tilde{C} = M_v Y^s$, where $s = z_3 + s = z_4$. Then $\mathcal{D}_l$ randomly selects $s' \xleftarrow{R} \mathbb{Z}_p$ and sets $C_\Delta = Y^{s'}$, $\hat{C}_0 = g_1^{s'}$, $C_1 = g^{z_2 z_4} = g_2^{s''}$, $\hat{C}_1 = Z = g^{z_1 z_3} = g_1^{s - s''}$, $C_x = g_2^{s' H(m)}$, where $s'' = z_4$. Then $\mathcal{D}_l$ selects $\{\sigma_{i,\Delta}, \sigma_{i,0}, \sigma_{i,1} \xleftarrow{R} \mathbb{G} | 1 \leq i \leq n\}$ such that $\prod_{i=1}^n \sigma_{i,\Delta} = \prod_{i=1}^n \sigma_{i,0} = \prod_{i=1}^n \sigma_{i,1} = 1_{\mathbb{G}}$, and computes the ciphertext component $\{\{C_{i,t,\Delta}, C_{i,t,0}, \hat{C}_{i,t,0}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}$ in $\mathbf{G}_l$ is the same as in $\mathbf{G}_{l-1}$ exception the coponents $\{C_{i_l,t_l,\Delta}, C_{i_l,t_l,0}, \hat{C}_{i_l,t_l,0}\}$. Here $C_{i_l,t_l,\Delta} = \sigma_{i_l,\Delta} g^{s' \tau_{i_l,t_l}}$, $C_{i_l,t_l,0} = \sigma_{i_l,0} (g^{z_2 z_4})^{a_{i_l,t_l}}$ and $\hat{C}_{i_l,t_l,0} = \sigma_{i_l,1} Z^{b_{i_l,t_l}}$. If $Z = g^{z_1 z_3}$, $\mathcal{A}$ is in game $\mathbf{G}_{l-1}$, otherwise $\mathcal{A}$ is in game $\mathbf{G}_l$. For *Phase 2*, the distinguisher does as in *Phase 1*. For *Guess*, $\mathcal{A}$ output a bit v'. If $v = v'$, $\mathcal{D}_l$ outputs 1, otherwise $\mathcal{D}_l$ outputs 0. In case 1, when $Z = g^{z_1 z_3}$, the view of $\mathcal{A}$ is distributed exactly as $\mathcal{A}'s$ view in $\mathbf{G}_{l-1}$. We have $\Pr[\mathcal{D}(g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{z_3+z_4})] = \Pr[\varepsilon_{l-1}]$. In case 2, Z is a random element in $\mathbb{G}_T$, the view of $\mathcal{A}$ is distributed exactly as $\mathcal{A}'s$ view in $\mathbf{G}_l$. We have $\Pr[\mathcal{D}(g, g^{z_1}, g^{z_2}, g^{z}, g^{z_2 z_4}, g^{z_3+z_4})] = \Pr[\varepsilon_l]$. So $|\Pr[\varepsilon_{l-1}] - \Pr[\varepsilon_l]| = |\Pr[\mathcal{D}(g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{z_3+z_4}) - \Pr[\mathcal{D}(g, g^{z_1}, g^{z_2}, g^{z}, g^{z_2 z_4}, g^{z_3+z_4})]| \leq \epsilon_{DL}$.

Theorem 9. *Our Multi-user searchable encryption scheme is $\mathcal{L}$-semantically secure against the adaptive attacks under the assumptions that DDH assumption holds in $\mathbb{G}$, F and F_p are secure PRFs and smABE is a CPA secure encryption.*

Proof. This proof is similar to [Sun et al. 2016]. The main difference contains two aspects. First, we use our smABE to encrypt the document identifiers *id* instead of CP-ABE. Since the smABE is IND-sCP-CPA secure, an adversary cannot distinguish whether a ciphertext is for the desired *id* or an arbitrary message. Second, in our scheme, the simulator also needs to construct mt and $D_{\Delta,0}$. It is easy to construct $D_{\Delta,0}$ with the information of $\mathbf{sk}_{mat}$. To construct mt, we use a

Table 1: Comparison Between Two Schemes

Schemes	Sun et al.'s scheme	Our Proposed Scheme
Access Policy Protection	No	Yes
Encryption Cost	$\|\mathcal{T}\| + \|\mathbb{G}_T\| + (2k+1)\|\mathbb{G}\|$	$2\|\mathbb{G}_T\| + (3\eta+4)\|\mathbb{G}\|$
Match Cost	-	$2\mu P$
Communication Cost	$\|\mathrm{DB}(\bar{\mathbf{w}})\| \cdot [\,\|\mathcal{T}\| + \|\mathbb{G}_T\| + (2k+1)\|\mathbb{G}\|\,]$	$l\,\|\mathrm{DB}(\bar{\mathbf{w}})\| \cdot [\,\|\mathbb{G}_T\| + 4\|\mathbb{G}\|\,]$
Decryption Cost	$l\,\|\mathrm{DB}(\bar{\mathbf{w}})\| \cdot [\,(2k+1)P + hE\,]$	$l\,\|\mathrm{DB}(\bar{\mathbf{w}})\| \cdot 4P$

random oracle to generate the auxiliary message m such that to keep consistent with the **SMSE.Setup** algorithm. Therefore, our proposed SMSE scheme is also $\mathcal{L}$-semantically secure against the adaptive attacks.

5.2 Comparison

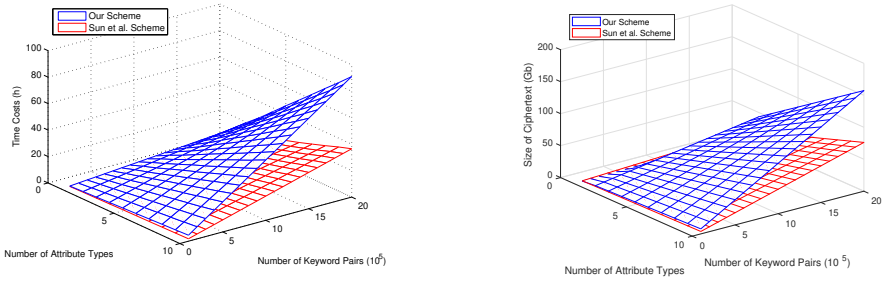
In this section, we compare the proposed scheme with Sun et al. scheme. Table 1 presents the comparison between these two schemes. We denote E as an exponentiation operation in $\mathbb{G}_T$, P as a computation operation of a paring, $|\mathcal{T}|$ as the size of the access policy tree $\mathcal{T}$, k as the number of attributes in the $\mathcal{T}$, h as the number of non-leaf node in $\mathcal{T}$, $|\mathrm{DB}(\bar{\mathbf{w}})|$ as the search results for the queried keywords $\bar{\mathbf{w}}$, $|\mathbb{G}|$ as the bit-length of an element in group $\mathbb{G}$, l as the ratio of the number of search results for the queried keywords to the number of ones which can be decrypted by the user.

5.3 Performance Evaluation

In this section, we provide the experimental evaluation of the proposed SMSE scheme. Our experiments are implemented with the pairing-based cryptography (PBC) library and OpenSSL open-source library on a LINUX machine with Intel Core i5-6500 CPU processors running at 3.20 GHz and 16 G memory. We evaluate the computation cost, storage cost and transmission cost between our proposed scheme and Sun et al. scheme [Sun et al. 2016].

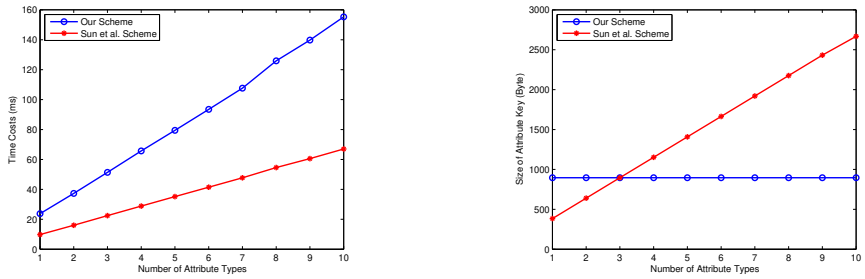
In the **Setup** algorithm, the main difference between our proposed scheme with the scheme in [Sun et al. 2016] is the generation of the ciphertext $\mathbf{e}$. In our scheme, the ciphertext $\mathbf{e}$ is generated by the proposed smABE, while $\mathbf{e}$ in [Sun et al. 2016] is generated by CP-ABE [Bethencourt et al. 2007]. It is first to generate the system public key pk and the system master key mk, which are not related to the system attribute set and the access policy in both schemes. The size of pk and mk in the proposed smABE and CP-ABE is 764 Byte, 1020 Byte respectively and the time cost is 0.95s, 1.23s respectively.

The search index TSet and XSet are generated by the data owner and sent to the cloud. For each keyword, there are two elements in the TSet and one element



(a) Efficiency Comparison (b) Storage Comparison

Figure 2: Setup in SMSE.



(a) Efficiency Comparison (b) Transmission Comparison

Figure 3: KenGen in SMSE.

in the XSet. The difference between our SMSE scheme and Sun et al. scheme is the generation of element $\mathbf{e}$. So we implement the costs of $\mathbf{e}$ and the results are shown in Figure 2. Here we set the number of attribute type from 1 to 10 and each type has two values. The access policy in both schemes is set to the same. The number of keyword-document pair varies from one hundred thousand to 2 million. From Figure 2a, the time cost to generate $\mathbf{e}$ in both schemes increases with the number of attribute type and keyword-document pair. The time cost in our scheme is larger than that of Sun et al. scheme. From Figure 2b, the size of $\mathbf{e}$ also increases with the number of attribute type and keyword-document pair, which in our scheme is larger than that of Sun et al. scheme. However, the generation of $\mathbf{e}$ is one-time cost, and it will be outsourced to the server. Therefore, both the time cost and storage cost in our scheme are acceptable.

The comparison of generating the attribute private key sk_L is shown in Figure 3. The time cost for both schemes increases with the number of attribute types, and it is slightly larger in our scheme. For the size of attribute private key, it

is a constant number in our scheme. However, it increases with the number of attribute types in Sun et al. scheme. In multi-user searchable encryption setting, the attribute private key needs to be transmitted to the different users, so the communication cost is saved in our proposed scheme.

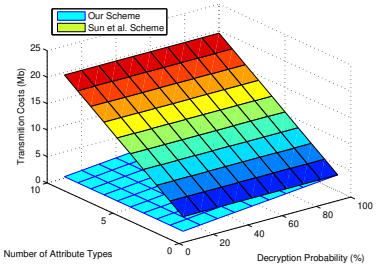


Figure 4: Transmission comparison.

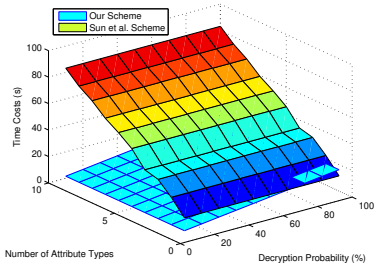


Figure 5: Efficiency comparison.

The comparison of the cost of transmitting the search results is shown in Figure 4. Here we set the number of search results is 5 thousands. For Sun et al. scheme, all the search results are needed to be transmitted to the user, no matter whether they can be decrypted. Besides, the size of ciphertext increases with the number of attribute types. For our proposed scheme, it just returns the ciphertext which can be decrypted by the user. Furthermore, the size of ciphertext is constant. From Figure 4, we can see the size of ciphertext in our scheme decreases with the decryption probability decreases and not increases with the number of attribute types. As a result, the transmission cost in our scheme is dramatically saved.

The comparison of the cost of decrypting the search results is shown in Figure 5. For Sun et al. scheme, the user needs to decrypt all the search results. In fact, it takes a little time to find out the ones which cannot be decrypted. However, the decryption cost for the ones which can be decrypted increases with the number of attribute types. However, in our scheme, the user just needs to decrypt the ones which is returned by the server. Besides, the decryption overhead is constant. As shown in Fig. 5, our decryption algorithm is very efficient.

6 Conclusion

In this paper, we proposed a new multi-user searchable encryption scheme that simultaneously achieves Boolean query and fast decryption. In our construction, a server-side match technique is proposed which can ensure the cloud server to test whether an encrypted data match the user's decryption ability. Furthermore,

we construct a new SMSE scheme with fast decryption based on the above technique. That is, the server can filter the search results which cannot be decrypted by the user. Formal security analysis and performance evaluation present that the proposed construction can achieve the desired goals and reduce the communication and decryption cost.

Acknowledgement

This work was supported by the National Key Research and Development Program of China (2017YFB0802202), National Natural Science Foundation of China (Nos. 61572382, 61702401), China 111 Project (No. B16037), Natural Science Basic Research Plan in Shaanxi Province (Nos. 2016JZ021, 2018JQ6001), and China Postdoctoral Science Foundation (Nos. 2017M613083).

References

[Ballard et al. 2005] Ballard, L., Kamara, S., Monrose, F.: "Achieving Efficient Conjunctive Keyword Searches over Encrypted Data"; Proc. of ICICS 2005, 10-13.

[Bethencourt et al. 2007] Bethencourt, J., Sahai, A., Waters, B.: "Ciphertext-Policy Attribute-Based Encryption"; 2007 IEEE Symposium on Security and Privacy (S&P). 2007 (May), 321–334.

[Bost et al. 2017] Bost, R., Minaud, B., Ohrimenko, O.: "Forward and Backward Private Searchable Encryption from Constrained Cryptographic Primitives"; Proc. of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS). 2017 (Oct), 1465–1482.

[Cash et al. 2013] Cash, D., Jarecki, S., Jutla, C. S., Krawczyk, H., Rosu, M., Steiner, M.:"Highly-Scalable Searchable Symmetric Encryption with Support for Boolean Queries"; Proc. of the 33^{th} Annual Cryptology Conference (CRYPTO). 2013 (Aug), 353–373.

[Chase 2007] Chase, M.:"Multi-authority Attribute Based Encryption"; Proc. of the 4^{th} Theory of Cryptography Conference (TCC). 2007 (Feb), 21–24.

[Chen et al. 2015a] Chen, X., Li, J., Huang, X., Ma, J., Lou, W.: "New Publicly Verifiable Databases with Efficient Updates"; IEEE Transactions on Dependable and Secure Computing, 12, 5 (2015), 546–556.

[Chen et al. 2016] Chen, X., Li, J., Weng, J., Ma, J., Lou, W.: "Verifiable Computation over Large Database with Incremental Updates"; IEEE Transactions on Computers, 65, 10 (2016), 3184–3195.

[Chen et al. 2014] Chen, X., Li, J., Ma, J., Tang, Q., Lou, W.: "New Algorithms for Secure Outsourcing of Modular Exponentiations"; IEEE Transactions on Parallel and Distributed Systems, 25, 9 (2014), 2386–2396.

[Chen et al. 2015b] Chen, X., Huang, X., Li, J., Ma, J., Lou, W., Wong, D. S.: "New Algorithms for Secure Outsourcing of Large-Scale Systems of Linear Equations"; IEEE Transactions on Information Forensics and Security, 10, 1 (2015), 69–78.

[Curtmola et al. 2006] Curtmola, R., Garay, J. A., Kamara, S., Ostrovsky, R.:"Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions"; Proc. of the 2006 ACM Conference on Computer and Communications Security (CCS). 2006 (Oct), 79-88.

[Goh 2003] Goh, E.:"Secure Indexes"; IACR Cryptology ePrint Archive, (2003), 216.

[Golle et al. 2004] Golle, P., Staddon, J., Waters, B. R.: "Secure Conjunctive Keyword Search over Encrypted Data"; Proc. of the 2^{nd} International Conference on Applied Cryptography and Network Security (ACNS). 2004 (Jun), 31-45.

[Goyal et al. 2006] Goyal, V., Pandey, O., Sahai, A., Waters, B.: "Attribute-based Encryption for Fine-grained Access Control of Encrypted Data"; Proc. of the 2006 ACM Conference on Computer and Communications Security (CCS). 2006 (Oct), 89-98.

[Jarecki et al. 2013] Jarecki, S., Jutla, C. S., Krawczyk, H., Rosu, M., Steiner, M.: "Outsourced Symmetric Private Information Retrieval"; Proc. of the 2013 ACM SIGSAC Conference on Computer and Communications Security (CCS). 2013 (Nov), 875-888.

[Kapadia et al. 2007] Kapadia, A., Tsang, P. P., Smith, S. W.: "Attribute-Based Publishing with Hidden Credentials and Hidden Policies'; Proc. of the Network and Distributed System Security Symposium (NDSS). 2007 (Feb), 179-192.

[Kamara et al. 2012] Kamara, S., Papamanthou, C., Roeder, T.: "Dynamic Searchable Symmetric Encryption"; Proc. of the 2012 ACM SIGSAC Conference on Computer and Communications Security (CCS). 2012 (Oct), 965–976.

[Kim et al. 2017] Kim, K. S., Kim, M., Lee, D., Park, J. H., Kim, W.: "Forward Secure Dynamic Searchable Symmetric Encryption with Efficient Updates"; Proc. of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS). 2017 (Oct), 1449–1463.

[Lai et al. 2011] Lai, J., Deng, R. H., Li, Y.: "Fully Secure Cipertext-Policy Hiding CP-ABE'; Proc. of the 7^{th} International Conference on Information Security Practice and Experience (ISPEC). 2011 (June), 24–39.

[Li et al. 2014] Li, J., Huang, X., Li, J., Chen, X., Xiang, Y.: "Securely Outsourcing Attribute-Based Encryption with Checkability"; IEEE Transactions on Parallel and Distributed Systems, 25, 8 (2014), 2201–2210.

[Li et al. 2018] Li, J., Zhang, Y., Chen, X., Xiang, Y.: "Secure Attribute-based Data Sharing for Resource-limited Users in Cloud Computing"; Computers & Security, 72 (2018), 2011 (May), 1–12.

[Li et al. 2019] Li, Y., Yu, Y., Susilo, W., Min, G., Ni, J., Choo, R.,: "Fuzzy Identity-Based Data Integrity Auditing for Reliable Cloud Storage Systems". IEEE Transactions on Dependable and Secure Computing, 16(1) (2019), 72-83.

[Liang et al. 2016] Liang, K., Huang, X., Guo, F., Liu, J. K.: "Privacy-Preserving and Regular Language Search Over Encrypted Cloud Data"; IEEE Transactions on Information Forensics and Security, 11, 10 (2016), 2365–2376.

[Nishide et al. 2008] Nishide, T., Yoneyama, K., Ohta, K.: "Attribute-Based Encryption with Partially Hidden Encryptor-Specified Access Structures"; Proc. of the 6^{th} International Conference on Applied Cryptography and Network Security (ACNS). 2008 (Jun), 111–129.

[Ryu and Takagi 2007] Ryu, E., Takagi, T.: "Efficient Conjunctive Keyword-Searchable Encryption"; Proc. of the 21^{st} International Conference on Advanced Information Networking and Applications (AINA). 2007 (May), 409–414.

[Song 2000] Song, D. X., Wagner, D., Perrig, A.: "Practical Techniques for Searches on Encrypted Data"; IEEE S&P. 2000 (May), 44–55.

[Song et al. 2018] Song X., Dong C., Yuan D., Xu Q., Zhao M.: "Forward Private Searchable Symmetric Encryption with Optimized I/O Efficiency"; IEEE Transactions on Dependable and Secure Computing, 2018.

[Stefanov et al. 2014] Stefanov, E., Papamanthou, C., Shi, E.: "Practical Dynamic Searchable Encryption with Small Leakage"; Proc. of the 21^{st} Annual Network and Distributed System Security Symposium (NDSS). 2014 (Feb), 23–26.

[Sun et al. 2016] Sun, S., Liu, J. K., Sakzad, A., Steinfeld, R., Yuen, T. H.: "An Efficient Non-interactive Multi-client Searchable Encryption with Support for Boolean Queries"; Proc. of the 21^{st} European Symposium on Research in Computer Security (ESORICS). 2016 (Sep), 26–30.

[Wang et al. 2015] Wang, J., Chen, X., Huang, X., You, I., Xiang, Y.: "Verifiable Auditing for Outsourced Database in Cloud Computing"; IEEE Transactions on Computers, 64, 11 (2015), 3293–3303.

[Wang et al. 2013] Wang, J., Ma, H., Tang, Q., Li, J., Zhu, H., Ma, H., Chen, X.: "Efficient Verifiable Fuzzy Keyword Search over Encrypted Data in Cloud Computing"; Computer Science and Information Systems, 10, 2 (2013), 667–684.

[Wang et al. 2017] Wang, J., Chen, X., Li, J., Zhao, J., Shen, J.: "Towards Achieving Flexible and Verifiable Search for Outsourced Database in Cloud Computing"; Future Generation Computer Systems, 67 (2017), 266-275.

[Wang et al. 2008] Wang, P., Wang, H., Pieprzyk, J.: "Keyword Field-Free Conjunctive Keyword Searches on Encrypted Data and Extension for Dynamic Groups"; Proc. of the 7^{th} International Conference on Cryptology and Network Security (CNS). 2008 (Dec), 178–195.

[Wang et al. 2016b] Wang, Y., Wang, J., Chen, X.: "Secure Searchable Encryption: a Survey"; Journal of Communications and Information Networks, 1, 4 (2016), 52-65.

[Wang et al. 2017] Wang, Y., Wang, J., Sun, S., Liu, J. K., Susilo, W., Chen, X.: "Towards Multi-user Searchable Encryption Supporting Boolean Query and Fast Decryption"; Proc. of the 11^{th} International Conference on Provable Security (ProvSec). 2017 (Oct), 24-38.

[Xue et al. 2019] Xue, L., Yu, Y., Li, Y., Au, M. H. Du, X., Yang, B.: "Efficient attribute-based encryption with attribute revocation for assured data deletion". Information Science, 479 (2019), 640-650.

[Yu et al. 2018A] Yu, Y., Li, Y., Du, X., Chen, R., Yang, G.: "Content Protection in Named Data Networking: Challenges and Potential Solutions". IEEE Communications Magazine 56(11) (2018), 82-87.

[Yu et al. 2017] Yu, Y., Au, M. H., Ateniese, G., Huang, X., Susilo, W., Dai, Y., Min, G.: "Identity-Based Remote Data Integrity Checking with Perfect Data Privacy Preserving for Cloud Storage". IEEE Transactions on Information Forensics and Security, 12(4) (2017), 767-778.

[Yu et al. 2019] Yu, Y., Li, Y., Yang, B., Susilo, W., Yang, G., Bai, J., "Attribute-Based Cloud Data Integrity Auditing for Secure Outsourced Storage": IEEE Transactions on Emerging Topics in Computing. doi:10.1109/TETC.2017.2759329.

[Yu et al. 2018] Yu, Y., Xue, L., Li, Y., Du, X., Guizani, M., Yang, B.: "Assured Data Deletion with Fine-Grained Access Control for Fog-Based Industrial Applications". IEEE Transactions on Industrial Informatics, 14(10) (2018), 4538-4547.

[Zhang et al. 2017] Zhang, Y., Chen, X., Li, J., Wong, D. S., Li, H. You, I.: "Ensuring Attribute Privacy Protection and Fast Decryption for Outsourced Data Security in Mobile Cloud Computing"; Information Sciences, 379 (2017), 42-61.

[Zuo et al. 2016] Zuo, C., Macindoe, J., Yang, S., Steinfeld, R., Liu, J. K.: "Trusted Boolean Search on Cloud Using Searchable Symmetric Encryption"; IEEE Trustcom, 2016, 113–120.