Efficient Attribute-Based Searchable Encryption on the Cloud Storage

Wanfen Guo, Xiaolei Dong, Zhenfu Cao, Jiachen Shen

August 16, 2017

Abstract

Cloud computing is very popular for its computing and storage capacity at lower cost. More and more data are being moved to the cloud to reduce storage cost. On the other hand, since the cloud is not fully trustable, in order to protect data privacy against third-parties and even the cloud server, they are usually encrypted before uploading. However, many operations, such as searching, are hard to perform on encrypted data. To solve this problem, searchable encryption has emerged. Searchable encryption in multi-user setting is much less efficient than in single-user setting. In order to address this problem, we propose a multi-owner to multi-user searchable encryption scheme based on attribute-based encryption. Our scheme keeps data secure in the cloud even against the cloud server. It allows users with appropriate authorizations to perform search operations on encrypted data. In addition, search tokens are generated by users instead of data owners. We prove that token privacy and index privacy are well protected in our scheme. The cloud server and illegimate users are not able to get any useful information about search tokens and ciphertexts. Ciphertexts of our scheme are constant-size, which reduce the time-complexity and bandwidth overhead of our scheme.

Keywords: Cloud storage, cloud computing, attribute-based encryption, multi-users searchable encryption

1 Introduction

With the development of the information technology, more and more information exchanges on the Internet. Therefore, not only the computers but also the mobile devices produce high-volume of data. As a result, cloud storage has become more and more popular with the low cost and vast capacity. While the information uploaded to the cloud may be sensitive and data owners want to keep them secret and prevent them from being exposed. What concerns data owners is that the cloud server is capable of manipulating their data in the cloud storage, so a method of encryption on the important data before uploading has been adopted. On the other hand, the goal of file storage is that we can find data when we need them. Although encryption helps in keeping data safe, it makes performing computation on the data difficult, such as search operations. One of naive methods is to download all encrypted files and decrypt them, then we can perform search operations on the plaintext files. This results in the huge bandwidth overhead and extra cost on the storage of downloaded files. The searchable encryption(SE) can solve the problem of searching on the ciphertext and take full advantage of the cloud computing. Let's consider a scenario: In a hospital, all patients' information are uploaded to the hospital system in the form of ciphertexts, which include their names, telephone numbers, disease types, record time, the name of their doctors and so on. As a patient, he should be able to search all the information about himself and nothing about other people. As a doctor, to track the patients' recovery, he can search the names and telephone numbers of his patients. Even the doctor of a patient A changed for some reason, the new doctor still can search the name and telephone number of A without encrypting A's information again. The statistical department can search the number of the patients who affected some type of diseases during a certain period of time. Therefore, in this scenario, the different authorities should be given to different people so that they can search different information according to their roles. However, most existing schemes are not able to achieve this. In the general symmetric searchable encryption(SSE) setting, when a data user wants to search on the files, he should request a search token from the data owner. After sending the search token to the cloud, the data users will receive search results computed by the cloud server. In this case, we can see that the keyword that the data user searched is known by the data owner which is not desirable. What's more, multi-owners to multi-users scheme model is more practical than one-owners to one users. And, the computing capacity of mobile phones and wearable devices are so week that they only perform efficient algorithms. The problem of the computation complexity and space complexity of is worth noting. And the response speed of search operations should be improved.

1.1 Our Contributions

We design an efficient attribute-based searchable encryption(EABSE) on the cloud storage scheme, which allows the cloud to execute the search operations on the ciphertext and forbids the cloud to get more information except the search results. We use inverted index data structure to store the index of the files. The index is encrypted using our EABSE schemes and we select a symmetric scheme(such as AES, etc.) to encrypt the files. Our contributions can be concluded as bellows:

Our scheme implements multi-owners to multi-users searchable encryption on the base of ciphertext-policy attribute-based encryption. The data owners use a certain access policy to encrypt the keywords in index, and the data users whose attributes satisfy the access policy can achieve the corresponding search tokens.

Our scheme protects the privacy of both the data owners and the data users. Only when data users' attributes satisfy the data's access policy, he can get the data which encrypted by the data owner. On the other hand, in this scheme, the search token is computed by the users, and the data owners don't know which word the users want to search. While in the general scenes, the data user must submit the keyword he want to search and apply the data owners for search tokens.

Under the random oracle security model, we define the security of the E-ABSE basing on the DL-assumption and construct a proper scheme. Moreover, we analyze the security of the scheme and prove it secure from the aspects of correctness and privacy.

We improve the efficiency of the scheme to meet the needs of the practical scenes. In time complexity, we improved the complex computation operations. In space complexity, we reduce the size of the ciphertext greatly.

The inverted index, the bilinear pairing and ciphertext policy attribute-based encryption are the powerful tools in the scheme.

1.2 Related Work

Many search schemes have been proposed after the symmetric searchable encryption was first introduced in [1] and asymmetric searchable encryption was introduced in [2].

Searchable Encryption SE technology solves the problem of searching on the encrypted files and improves the practicability of cloud storage and cloud computing. Not only the functionality but also efficiency of the searchable encryption have improved a lot. In [3], the scheme which supported searching multi-keyword at the same time was proposed. [5] introduced a searchable scheme with dynamic updating. And, the time of updating, which includes addition and deletion, was as much as searching. To improve the efficiency, utilizing the useful RAM as a solution like [6] was deployed. Since Abdalla et al. [4] constructed asymmetric searchable encryption scheme from identity based encryption(IBE) and proved it secure, some works transformed the efficient IBE into efficient PEKS [23]. The multi-participants searchable encryption allowed data share among many people [21, 22], where the authorised data-users can search on the files uploaded by according data-owners. Our work emphasises the research on multi-users model in searchable schemes.

Index The index of the files is very important in search. Different indexes have both advantages and disadvantages. Curtmola et al. [1] proposed the first encrypted searchable index basing on the inverted index. Some searches base on inverted index because of the efficiency while it is not convenient for the files updating. The index basing on a bloom filter was introduced by Goh et al. [7]. While Chang et al. proposed a vector index in [8]. Diverse kinds of index are proposed to assist in perfecting the scheme.

Attribute-Based Searchable Encryption To remove the Trusted Authority(TA) in identity-based encryption schemes, Sahai and Waters [15] presented FIBE as a solution which we considered as the prototype of Attribute-Based Encryption(ABE). In practical scenes, ciphertext-policy ABE(CP-ABE) as an effective method to solve the data share in safety between multiple data-owners and multiple data-users, was very popular and discussed by many papers [16–18]. ABE family is diversified a lot. ABE plays an important role in fine-grained access control. The searchable encryption scheme with data users' attributes as the search secret key is more practical. Attribute-based searchable encryption(ABSE) and its application were proposed in [14,24]. As scheme in [14], data owners encrypted their index using different access policy, then data users were able to search keywords that they were interested in if their attributes satisfied the associated access policy. Besides, [14] proposed verifiable attribute-based searchable encryption on the malicious cloud. Most existing works has implemented additional function of ABE in ABSE, like revocability of data users' search right. However, the computation complexity and communication efficiency of the above schemes were not taken into account. In this paper, we use ABE as a smoothly tools to implement the search on ciphertext. Therefore, we require the ABE scheme with enough efficiency. We adopt constant-ciphertext extension ABE [19, 20] to prevent the time complexity and space complexity increase obviously with the data growing on the cloud. Besides, we reduce the communication cost between data-owners and cloud when upload the constant ciphertext.

2 Preliminary

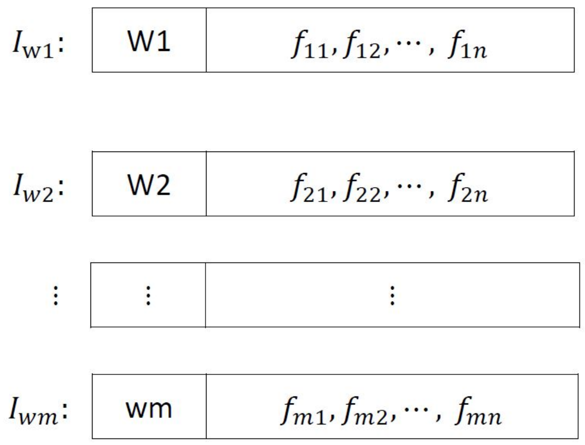
Inverted Index In the searchable encryption systems, the inverted index [10] is a practical data structure. As showed in the Figue1, the inverted index includes many inverted lists $I_{w_1}, I_{w_2}, \cdots, I_{w_m}$, where m is the size of the keyword dictionary. One inverted list I_{w_i} contains all files that contain the keyword w_i. The files $f_1, f_2, \cdots, f_{n1}$ in I_{w_1} include the keyword w_1 where n1 denotes the number of the files. The reason that we use inverted index like most works [11–13] is the search efficiency can be improved comparing to other indexes [1,9].

Symmetric Bilinear Group Let G and G_T be the two cyclic multiplicative group of prime order p, and g is the generator of G. There exists a pairing e: $G \times G \longrightarrow G_T$ with the below three properties:

(1) Bilinearity: $e(g^a, g^b) = e(g, g)^{ab}$, for $a, b \in Z_p$;

(2) Non-degeneracy: $e(g, g) \neq 1$;

(3) Computability: for all $a, b \in Z_p, e(g^a, g^b)$ is efficiently computed.



I_{w1}:
W1
$f_{11}, f_{12}, \cdots, f_{1n}$
I_{w2}:
W2
$f_{21}, f_{22}, \cdots, f_{2n}$
I_{wm}:
wm
$f_{m1}, f_{m2}, \cdots, f_{mn}$

Figure 1: Inverted Index

Decisional Linear (DL) Assumption There exists g, f, h chosen randomly from the group G and r_1, r_2 chosen randomly from Z_p. DL assumption says the probability of a probabilistic polynomial-time algorithm $\mathcal{A}$ can distinguish $Q = h^{r_1+r_2}$ with Q is random element from the tuple $(g, f, h, f^{r_1}, g^{r_2}, Q)$ is negligible. We define ϵ is a negligible parameter, then the advantage of $\mathcal{A}$ in solving DL problem is

$$|Pr[\mathcal{A}(g, f, h, f^{r_1}, g^{r_2}, h^{r_1+r_2}) = 1] - Pr[\mathcal{A}(g, f, h, f^{r_1}, g^{r_2}, Q) = 1]| \leqslant \epsilon$$

3 Our Scheme

The scheme implements the multi-owners to multi-users keyword search functionality with high-efficiency. In order to satisfy the requirement of real environment, we promise the correctness of the search result but also the privacy of the data in storage and transmission.

3.1 System model

There are three main entities: data owner, data user and cloud server. Following the figure2, data owners encrypt their files $F = \{f_1, f_2, \cdots, f_n\}$ into the ciphertexts $C = \{c_1, c_2, \cdots, c_n\}$ using some symmetric encryption. Because the file encryption is common, we don't discuss it in detail in this paper. Of course, a pre-defined keyword dictionary $WD = \{w_1, w_2, \cdots, w_m\}$ is needed. Every data owner constructs his encrypted inverted index according to the keyword dictionary. We can use the identifier of the file instead of the file itself stored in the index for convenience in actual application. Finally, data owner uploads all the above information to cloud server. When the data user wants to search

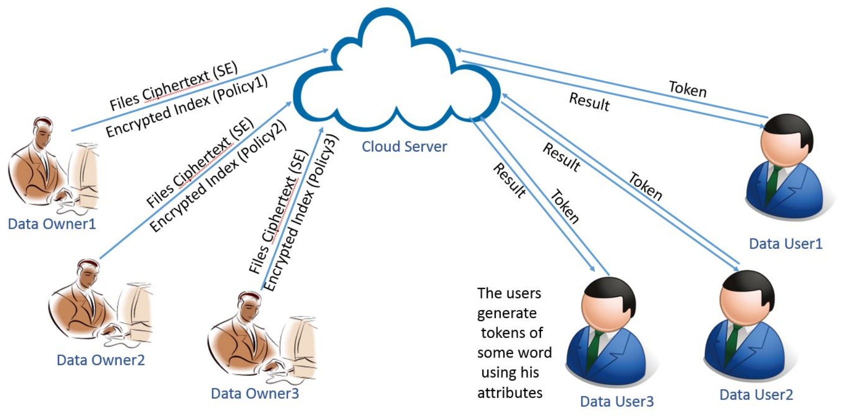


Figure 2: System Model

some keyword, he generates search token using his attributes. The cloud server executes the search on the index after receiving the token and returns the result to the user.

3.2 Algorithms and Security

3.2.1 Algorithms

According to the system model, we divide our scheme into three parts: data upload, query and data download. Each part is implemented by several algorithms respectively.

Data Upload This part is executed by the data owner and system. The owner generates a secure inverted index and encrypts files before uploading.

$(msk, pk) \leftarrow Setup(U, 1^{\lambda})$. On input a security parameter λ and the attribute set U(which includes the all attributes of users), this probabilistic algorithm outputs the master key msk and public key pk.

$(cphW, cphF) \leftarrow Enc(S, WD, F, pk)$. After receiving the access policy S, keyword dictionary $WD = \{w_1, w_2, \cdots, w_m\}$ and the file set $F = \{f_1, f_2, \cdots, f_n\}$, the probabilistic encryption algorithm generates the encrypted index and the ciphertext of files. The file encryption is executed using some simple symmetric encryption considering of the efficiency.

Query This part is executed by the data user and the cloud server. The user generates token for the keyword he wants to query.

$sk \leftarrow KeyGen(Attr, msk, pk)$. Given the attributes $Attr$ by the user, master key msk and public key pk, the probabilistic algorithm outputs the secret key sk.

$tok \leftarrow TokenGen(pk, sk, w)$. This algorithm is used to generate token tok with the secret key sk and the keyword w to be queried.

Data Download This part is executed by the cloud server. After receiving the search token, the cloud server runs Search algorithm on the data in cloud storage. Then it downloads the matched ciphertexts of files and returns them to user.

$rslt \leftarrow Search(tok, cphW, cphF)$. This deterministic algorithm outputs the ciphertext of the files $cphF$ if $cphW$ can matched with the tok, outputs $\perp$ otherwise.

3.2.2 Security

In the general schemes, we suppose the data owner and the data user are trusted, but the cloud server is semi-trusted. It means that the cloud server would execute data users' instructions honestly but try to get the information about what it is curious about. In the common ABE schemes, the data is closed absolutely after encryption. While the ciphertexts in searchable encryption must leave a threshold for each keyword ciphertext. Therefore, the security of the searchable encryption is more tedious. In this paper, we guarantee the scheme secure from two aspects: correctness and privacy.

Correctness The search algorithm outputs the right result if only if the inputs of the $Search(token, cphW, cphF)$ algorithm are from the $Setup$, $KeyGen$, Enc and $TokenGen$. We say the scheme is correct when the above statement holds.

Privacy For the files, the ciphertexts $cphF$ are semantic security by adopting a symmetric encryption, such as AES, then the adversary can not get information from what the file ciphertext revealed with non-negligible probability. For the keyword, we have:

- Index Privacy: A probabilistic polynomial-time adversary can not get any useful information from the encrypted keyword cphW without associate search tokens. That is, the adversary can not determine which of the ciphertext of the keyword w_1 and the ciphertext of the keyword w_2 is about the keyword w he submitted.
- Token Privacy: Given the keyword token, a probabilistic polynomial-time adversary can not learn the keyword plaint. In other words, the cloud server can not learn which keyword the data user queried.

Our scheme implements a secure searchable encryption if it satisfies all above requirements.

3.2.3 Threat Model

Under the random oracle model, we prove the index privacy of the EABSE scheme with selectively chosen-keyword attack game [14] and prove the token privacy with token privacy game.

- Selectively Chosen-Keyword Attack(SCKA) Game

  Setup: At first, the adversary gives a access policy S to the challenger. The challenger plays as the system and runs $Setup(U, 1^{\lambda})$, then it keeps the master key mk.

  Query Phase1: For the efficiency, the challenger applies for a query list ql and sets it empty before the Phase1 running. The adversary adaptively queries the below for polynomial times:
  With the $KeyGen(Attr, msk, pk)$ algorithm, the adversary inputs the attributes $Attr$ to challenger and gets associated secret key sk from the challenger if the attributes can not satisfy access policy S, otherwise the procedure aborts; according to the $TokenGen(pk, sk, w)$ algorithm, the adversary can get a search token after inputting sk and keyword w. The challenger puts w into the querylist ql if the token is legitimate for the keyword.

  Challenge phase: The adversary selects two keywords w_0 and w_1 randomly which are excluded from the list ql. The challenger flips a coin to select $\mu \leftarrow \{0, 1\}$ and then runs the Enc algorithm. The challenger returns the ciphertext cph^* of the w_μ to the adversary.

  Query Phase 2: The adversary can execute the queries as Phase1 did but the keyword w_0 and w_1 can not be queried any more.

  Guess: Finally, adversary gives a guess $\mu^{'}$ of μ and wins the game if $\mu^{'} = \mu$.
  We can define the advantage of adversary winning the game is $|Pr[\mu^{'} = \mu] - \frac{1}{2}|$. If the advantage is negligible, our scheme is secure with index privacy.

- Token Privacy Game

  Setup: The challenger plays as the system and runs $Setup(U, 1^{\lambda})$, then it keeps the master key msk.

  Query Phase 1: As in SCKA game, the challenger selects the key list kl which is empty initially. The adversary can execute below process for polynomial times. The adversary chooses a attribute set $Attr$ as the input of $KeyGen$ algorithm. The challenger returns KeyGen's output sk to adversary and add $Attr$ into kl. Then, challenger runs $TokenGen$ algorithm and returns $token$ to adversary after receiving the sk and keyword w.

  Challenge phase: The adversary submits a access policy S with the restriction that each $Attr$ in kl can not satisfy S. The challenger randomly chooses a keyword w^* from the keyword dictionary and encrypt it with the

S into $cphW^*$. The challenger selects a attribute set $Attr^*$ which satisfies S. The adversary gets $token^*$ from challenger.

Query Phase 2: As did in Phase 1 except that keyword w^* can not be queried.

Guess: The adversary outputs guess keyword $w^{'}$, and can win the game if $w^{'} = w^*$.

3.3 Constructions

In our scheme, we use inverted index structure as introduced above and implement searchable encryption with AND gate as access control. The scheme consists of 5 main algorithms. We introduce them in detail as below:

Init We suppose all the attributes are included in set U=$\{attr_1, attr_2, \cdots, attr_n\}$, where n is the size of U. For each attribute $attr_i (1 \leq i \leq n)$, there has 2 values v_i and $\neg v_i$. If the attributes set $Attr$ of one data user include attribute $attr_i (1 \leq i \leq n)$, the value of $attr_i$ is v_i and the value of $attr_i$ is $\neg v_i$ if $attr_i$ is not in $Attr$. To formalize the description of attributes, we adopt the value of attribute to represent whether user's set contains this attribute.

Setup Given a bilinear group $e : G \times G \rightarrow G_T$, p as prime order of G and G_T, and $H : \{0,1\}^* \rightarrow Z_p$ as an one-way hash function, randomly select three numbers $a, b, c \leftarrow Z_p$, a set $\{r_1, r_2, \cdots, r_{2n}\} \leftarrow Z_p$ and a set $\{x_1, x_2, \cdots, x_{2n}\} \leftarrow G$. Set $u_i = g^{-r_i}$ and $y_i = e(x_i, g)$, where $1 \leq i \leq 2n$. Then, output the public key $pk = (g, g^a, g^b, g^c, (u_i, y_i) | 1 \leq i \leq 2n)$ and the master key $msk = (a, b, c, (r_i, x_i) | 1 \leq i \leq 2n)$.

Enc Choose random $t_1, t_2 \in Z_p$. Suppose the access policy structure $S = \bigwedge_{v_i \in U} v_i^{'}$, where $v_i^{'} = v_i$ or $\neg v_i$. Set $u_i^{'} = u_i$ if $v_i^{'} = v_i$, $u_i^{'} = u_{i+n}$ otherwise. Compute $u_{gate} = g^{t_2} \prod_{i=1}^{n} u_i^{'}$. For each keyword $w \in WD$, then set $W^{'} = g^{ct_1}$, $W = g^{a(t_1+t_2)} g^{bH(w)t_1}$, and encrypt files F which associate with the keyword w with some symmetric encryption algorithm into $cphF$. Obviously, $cphW = (W^{'}, W, u_{gate})$. Then, the whole $cph = (cphW, cphF)$ as the result of encryption.

KeyGen At First, we set $v = g^{ac}$. For each attribute v_i^* in data user's attribute collection, set $y_i^* = y_i$ if $v_i^* = v_i$, $y_i^* = y_{i+n}$ otherwise. Similarly, compute $\sigma_i^* = x_i v^{r_i}$ if $v_i^* = v_i$, $\sigma_i^* = x_{i+n} v^{r_{i+n}}$ otherwise. Set $\sigma_{user} = \prod_{i=1}^{n} \sigma_i^*$ Then, the secret key $sk = (y_{user} = \prod_{i=1}^{n} y_i^*, < v, \sigma_{user} >)$.

TokenGen Select $s \leftarrow Z_p$. To generate the search token for keyword w, compute $tok1 = (g^a g^{bH(w)})^s$, $tok2 = g^{cs}$. Therefore, the search token $tok = (y_{user}^s, < v^s, \sigma_{user}^s >, tok1, tok2)$.

Search At first, compute E $= \frac{e(u_{gate}, v^s)e(\sigma^s_{user}, g)}{y^s_{user}}$. If user's attributes satisfy the access policy according to the ciphertext, E $= e(g,g)^{acst_2}$ and $e(W^{'}, tok1)E = e(W, tok2)$ holds.

According to the above, the search token *tok* can match with the *cphW* in the ciphertext *cph*, if the attributes of data user can satisfy the access policy used for encrypting the keyword. The *cphF* in *cph* should be downloaded afterwards and returned to the data user.

4 Security Analysis

In this part, we will introduce the security of the EABSE scheme. From the scheme construction, the correctness is easy to prove. Then, we show that the scheme meets the security definition in Privacy section.

Theorem 4.1. *If an adversary $\mathcal{A}$ wins the SCKA game mentioned above with the advantage ϵ, there is a Challenger $\mathcal{B}$ can break the DL assumption with the advantage $\frac{\epsilon}{2}$.*

Proof: **Setup:** The challenger run $Setup(U, 1^\lambda)$, randomly selects $\{r_1, r_2, \cdots, r_{2n}\} \leftarrow Z_p$ and $\{x_1, x_2, \cdots, x_{2n}\} \leftarrow G$, sets $g^a = h$, $g^c = f$, and $g^b = f^d$ where $d \leftarrow Z_p$. As in the scheme, $u_i = g^{-r_i}$ and $y_i = e(x_i, g)|i = 1, 2, \cdots, 2n$. It makes the public key $pm = (h, f^d, f, r_1, r_2, \cdots, r_{2n}, x_1, x_2, \cdots, x_{2n})$, then it keeps the master key $mk = (d, r_1, r_2, \cdots, r_{2n}, x_1, x_2, \cdots, x_{2n})$.

The adversary gives a access policy S to the challenger which it wants to challenge.

Query Phase 1: For the efficiency, the challenger applies for a query list ql and sets it empty before the phase running. The adversary adaptively queries the below for polynomial times:

The adversary inputs the attributes set *Attr* to challenger. If *Attr* satisfies the S, then the algorithm terminates. Select κ_{sec} as one of master key a. For each attribute in U according to *Attr*, computes σ_i and y_i. Set $v = f^{\kappa_{sec}}, \sigma_{user} = \prod_{i=1}^{n} \sigma_i$ where $\sigma_i = x_i v^{r_i}$, and $y_{user} = \prod_{i=1}^{n} y_i$. So $\mathcal{A}$ gets associated secret key $sk = (y_{user}, < v, \sigma_{user} >)$ from the challenger.

According to the $TokenGen(sk, w)$ algorithm, after choosing a random s$\leftarrow Z_p$, the adversary can get a search token $tok = (tok1 = (hf^{dH(w)})^s, tok2 = f^s, < v^s, {\sigma_{user}}^s >, {y_{user}}^s)$ after inputting sk and keyword w. The challenger put w into the querylist ql if the token is legitimate for the keyword.

Challenge phase: The adversary selects two keywords w_0 and w_1 randomly except from the list ql. The challenger flips a coin to select $\mu \leftarrow \{0, 1\}$ and then runs the Enc algorithm. Select $t_1, t_2 \leftarrow Z_p$, $\mathcal{B}$ computes $u_{gate} = g^{t_2} \prod_{i=1}^{n} u_i$, and $w^{'} = f^{t_1}, w = Qf^{dH(w_\mu)t_1}$. The challenger returns the ciphertext $cphW^* = (w^{'}, w, u_{gate})$ of the w_μ to the adversary. Because the access policy is public, it is easy to get g^{t_2} from the u_{gate}. Therefore, all the parameters in ciphertext are from the instance of the DL assumption.

Query Phase 2: The adversary can execute the queries as Phase1 did but the keyword w_0 and w_1 can not be queried.

Guess: Finally, adversary gives a guess $\mu^{'}$ of μ. If $\mu^{'} = \mu$, the challenger deems Q $= h^{r_1+r_2}$ implicitly. Because $cphW^*$ is a valid ciphertext for keyword w_μ only when Q $= h^{r_1+r_2}$ and w in $cphW^*$ is a random element of G when Q$\neq h^{r_1+r_2}$.

Therefore, we suppose adversary $\mathcal{A}$ can break the scheme with a non-negligible advantage ϵ, then the probability $\Pr[\mu^{'} = \mu]$ is $\frac{1}{2} + \epsilon$ if Q $= h^{r_1+r_2}$, $\Pr[\mu^{'} = \mu]$ is $\frac{1}{2}$ otherwise. Finally, the advantage of challenger solving the DL problem is $\frac{1}{2} \times (\frac{1}{2} + \epsilon) + \frac{1}{2} \times \frac{1}{2} - \frac{1}{2} = \frac{\epsilon}{2}$.

Theorem 4.2. *The challenger can break the Token Privacy Game with a negligible advantage if the probability of getting w from H(w) is negligible ν.*

Proof: **Setup:** The challenger plays as the system and runs $Setup(U, 1^\lambda)$ as in the scheme, then it makes the $pk = (g^a, g^b, g^c, (u_i, y_i)|1 \leq i \leq 2n)$ public and keeps the master key $msk = (a, b, c, (r_i, x_i)|1 \leq i \leq 2n)$ where $a, b, c, \{r_i|i = 1, 2, \cdots, 2n\} \leftarrow Z_p, \{x_i|i = 1, 2, \cdots, 2n\} \leftarrow G$ and $u_i = g^{-r_i}, y_i = e(x_i, g)$.

Query Phase 1: As in SCKA game, the challenger selects the key list kl which is empty initially.The adversary can execute below process for polynomial times qt. The adversary chooses a attribute set $Attr$ as the input of $KeyGen$ algorithm. The challenger returns KeyGen's output sk to adversary and add $Attr$ into kl. Then, challenger runs TokenGen algorithm and returns token $tok = (g^s, y^s_{user}, u^s_{user}, tok1, tok2)$ to adversary after receiving the sk and keyword w. (The marks such as $tok1$, $tok2$ are the same as the scheme.)

Challenge phase: The adversary submit a access policy S with the restriction that each $Attr$ in kl can not satisfy S. The challenger randomly chooses a keyword w^* from the keyword dictionary and encrypt it with the S into $cphW^*$. The challenger selects a attribute set $Attr^*$ which satisfies S. The adversary gets $token^*$ from challenger.

Query Phase 2: As did as Phase 1 with the restriction that the keyword w^* can not be queried.

Guess: The adversary $\mathcal{A}$ outputs guess keyword $w^{'}$, and can win the game if $w^{'} = w^*$. Simply, we can judge $\mathcal{A}$ wins the game if Search$(token^*, cphW^{'})$ returns the correct file result.

If the adversary wants to get the information about the keyword, it will analyse the tok1=$(g^a g^{bH(w)})^s$ in the token. The s is randomly chosen by the challenger, so the adversary can get information from the $H(w)$ leakage at most. If the probability of getting w from $H(w)$ is negligible ν, then adversary can break the Token Privacy game with $\frac{1}{|WD|-qt} + \nu$. The $|WD|$ denotes the size of the keyword dictionary WD and qt represents the query time in Phase 1. In practical scene, $|WD|$ is big enough. Finally, our scheme can gurantee the token privacy.

From the Construction section, we can prove the correctness of our proposed scheme. Combining with the above proofs, we can say the scheme satisfies the Index Privacy and Token Privacy. In general, the EABSE scheme is secure.

5 Performance and Efficiency Analysis

In the practical scenes, the users prefer the scheme with quicker response and less bandwidth cost. Therefore, we considered the efficiency of the scheme when we designed the scheme. In our scheme, we achieved the constant-size ciphertext. Basing on the computation on the group of prime order p, we mainly evaluate the exponentiation operation, multiplication operation and pairing operation in time complexity and the size of group G and G_T in space complexity. It is worth mentioning that multiplication operation is much more efficient than the exponentiation.

Compared to the CP-ABKS works in [14], we analyse the efficiency of the scheme from the two aspects of time complexity and space complexity. The results are showed in below tables Table1 and Table2. E denotes the exponentiation operation on the element in group G, E_T denotes the exponentiation operation on the element in group G_T. Similarly, M denotes the multiplication operation on the element in group G, M_T denotes the multiplication operation on the element in group G_T. And Pair is the symbol of the pairing operation. We use $|G|$ and $|G_T|$ as the remarks of the size of G and G_T respectively. At last, we use N to represent the number of attributes which satisfy the access policy and S to represent the number of attributes which owned by the data-user. (In our scheme, S=N.)

	Our Scheme	[14]
Enc	4E+(N+2)M	(2N+4)E+M
KeyGen	(S+1)E+2SM+SM_T	(2S+2)E+SM
TokenGen	6E+M	(2S+4)E+M
Search	4Pair+E_T+3M_T	(2N+3)Pair+NE_T+(N+2)M_T

Table 1: Time Complexity Analysis

	Our Scheme	[14]
Enc	3$\|G\|$	(2N+3)$\|G\|$
KeyGen	2$\|G\|$+$\|G_T\|$	(2S+1)$\|G\|$
TokenGen	4$\|G\|$+$\|G_T\|$	(2S+3)$\|G\|$

Table 2: Space Complexity Analysis

Considering the balance of time complexity and space complexity, our scheme aggregated $\sigma_1, \sigma_2, \cdots, \sigma_n$ into σ_{user} and $y_1, y_2, \cdots, y_n$ into y_{user} in *KeyGen* phase. This behavior reduced the sizes of secret key and token, thus saving the bandwidth cost. There is a transformation of our scheme. We can transmit $\sigma_1, \sigma_2, \cdots, \sigma_n$ and $y_1, y_2, \cdots, y_n$ as parts of the secret key. Then the cloud server should firstly aggregate the tokens σ_i^s, y_i^s in the *Search* phase. The computation works is transferred to the cloud server from the client. However, the communication cost between cloud server and the data user rise.

6 Conclusion

We introduced an efficient searchable scheme basing on the ciphertext-policy attribute-based encryption. The EABSE scheme allows secure data share in multi-owners and multi-users system model. The data owners upload the data wrapped with a certain access policy, and others can not get any information about the data whose attributes can not satisfy the access policy. While the authenticated data users are able to search a keyword w on the ciphertexts and get the encrypted files including w. Besides, the search token is generated by data users. This action prevents the keyword that users queried from being known by owners and improves the users' privacy. It is mentioned that, our scheme raises the efficiency of computation and reduces the cost of communication because of the constant-size ciphertext. Our performance and efficiency analysis illustrates this point effectively. The new requirements also rises with the times, so the efficient search on the dynamic dataset is still for future work.

References

[1] Curtmola, J. Garay, S. Kamara and R. Ostrovsky, Searchable symmetric encryption: improved denitions and efficient constructions. Proc, ACM CCS, 2006, pp. 79-88.

[2] D. Boneh, G.D. Crescenzo, R. Ostrovsky and G. Persiano, Public key encryption with keyword search. Proc, EUROCRYPT, May 2004,pp. 506-522.

[3] P. Golle, J. Staddon, B. Waters, Secure conjunctive keyword search over encrypted data. ACNS 2004. Lecture Notes in Com-puter Science, vol 3089. Springer, Berlin, Heidelberg.

[4] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, Searchable encryption revisited: consistecy properties, relation to anony-mous IBE and extentions. Springer, 2004.

[5] S. Gajek, Dynamic Symmetric Searchable Encryption from Constrained Functional Encryption. In: Sako K. (eds) Topics in Cryptology-CT-RSA 2016. Lecture Notes in Computer Science, vol 9610. Springer,Cham, pp. 75-89.

[6] S. Garg, P. Mohassel, C. Papamanthou, Efficient Oblivious RAM in Two Rounds with Applications to Searchable Encryption. In:Robshaw M., Katz J. (eds) Advances in Cryptology C CRYPTO 2016.CRYPTO 2016. Lecture Notes in Computer Science, vol 9816. Springer,Berlin, Heidelberg

[7] E. Goh, Secure Indexes. In: IACR Cryptology ePrint Archive, vol 2003

[8] YC. Chang, M. Mitzenmacher, Privacy Preserving Keyword Searches on Remote Encrypted Data. In: Ioannidis J., Keromytis A., Yung M. (eds)

Applied Cryptography and Network Security. ACNS 2005. Lecture Notes in Computer Science, vol 3531. Springer, Berlin, Heidelberg.pp, 442-455.

[9] C. Rongmao, M. Yi, Y. Guomin, G. Fuchun, H. Xinyi, W. Xiaofen, W. Yongjun, "Server-Aided Public Key Encryption With Keyword Search", Information Forensics and Security IEEE Transactions on, ISSN 1556-6013.vol. 11, 2016, pp. 2833-2842.

[10] D.E. Knuth, The art of computer programming,volume 1:Fundamental algorithms,2nd edition.Addison-Wesley (1973)

[11] S. Ji, On the Correctness of inverted index based public-key Searchable Encryp-tion scheme for Multi-time Search, 2016.

[12] R. Zhang, R. Xue, T. Yu, L. Liu, Dynamic and Efficient Private Keyword Search over Inverted Index–BasedEncrypted Data[J], ACM Transactions on Internet Technology (TOIT), 2016.

[13] B. Wang, W. Song, W. Lou, YT. Hou ,Inverted Index Based Multi-Keyword Public-key Searchable Encryption with Strong Privacy Guarantee, IEEE, 2015.

[14] Q. Zheng, S. Xu, G. Ateniese, Verifiable Attribute-based Keyword Search over Outsourced Encrypted Data, 2014 proceedings IEEE, 2014.

[15] A. Sahai and B. Waters. Fuzzy Identity Based Encryption. In Advances in Cryptology - Eurocrypt volume 3494 of LNCS pages 457-473. Springer 2005.

[16] L. Ibraimi, Q. Tang, PH. Hartel, W. Jonker, Efficient and Provable Secure Ciphertext-Policy Attribute-Based Encryption Schemes, Springer, 2009.

[17] B. Waters, Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient,and Provably Secure Realization, Springer, 2011.

[18] G. Yu, Z. Cao, G. Zeng, W. Han, Accountable Ciphertext-Policy Attribute-Based Encryption Scheme Supporting Public Verifiability and Nonrepudiation, International Conference on Provable Security. Springer International Publishing, 2016: 3-18.

[19] K. Zhang, J. Gong, S. Tang, J. Chen, X. Li, H. Qian, Practical and Efficient Attribute-Based Encryption with Constant-Size Ciphertexts in Outsourced Verifiable Computation, 2016.

[20] C. Chen, Z. Zhang, D. Feng, Efficient Ciphertext Policy Attribute-Based Encryption with Constant-Size Ciphertext and Constant Compution-Cost, Springer, 2011.

[21] Zhao, F., Nishide, T., Sakurai, K.: Multi-user keyword search scheme for secure data sharing with fine-grained access control. In: Kim, H. (ed.) ICISC 2011. LNCS, vol. 7259, pp. 406C418. Springer, Heidelberg (2012)

[22] X. Liu, Y. Zhang, B. Wang, J. Yan, Mona: secure multi-owner data sharing for dynamic groups in the cloud. IEEE Trans. Parallel Distrib. Syst. 24(6), 1182C1191 ,(2013)

[23] XA. Wang, F. Xhafa, W. Cai, J. Ma, F. Wei, Efficient Privacy Preserving Predicate Encryption with Fine-grained Searchable Capability for Cloud Storage, 2016.

[24] W. Sun, S. Yu, W. Lou, YT. Hou, Protecting Your Right: Attribute-based Keyword Search with Fine-grained Owner-enforced Search Authorization in the Cloud, IEEE, 2014.