Received 1 March 2015; revised 30 June 2015; accepted 2 October 2015. Date of publication 23 December 2015;
date of current version 2 March 2018.

Digital Object Identifier 10.1109/TETC.2015.2511457

Personalized Search Over Encrypted Data With Efficient and Secure Updates in Mobile Clouds

HONGWEI LI[1,2], (Member, IEEE), DONGXIAO LIU[1], (Student Member, IEEE), YUANSHUN DAI[1], (Member, IEEE), TOM H. LUAN[3], (Member, IEEE), AND SHUI YU[3], (Senior Member, IEEE)

[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[3]School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia

CORRESPONDING AUTHOR: H. LI (hongwei.uestc@gmail.com)

This work is supported in part by the State Key Laboratory of Information Security Open Foundation under Grant 2015-MS-02, in part by the China Post-Doctoral Science Foundation Funded Project under Grant 2014M552336 and Grant 2015T80972, in part by the International Science and Technology Cooperation and Exchange Program of Sichuan Province, China, under Grant 2014HH0029, and in part by the National Natural Science Foundation of China under Grant 61472065, Grant 61350110238, Grant U1233108, and Grant U1333127.

ABSTRACT Mobile cloud computing has been involved as a key enabling technology to overcome the physical limitations of mobile devices toward scalable and flexible mobile services. In the mobile cloud environment, searchable encryption, which enables direct search over encrypted data, is a key technique to maintain both the privacy and usability of outsourced data in cloud. On addressing the issue, many research efforts resolve to using the searchable symmetric encryption (SSE) and searchable public-key encryption (SPE). In this paper, we improve the existing works by developing a more practical searchable encryption technique, which can support dynamic updating operations in the mobile cloud applications. Specifically, we make our efforts on taking the advantages of both the SSE and SPE techniques, and propose PSU, a Personalized Search scheme over encrypted data with efficient and secure Updates in mobile cloud. By giving thorough security analysis, we demonstrate that the PSU can achieve a high security level. Using extensive experiments in a real-world mobile environment, we show that the PUS is more efficient compared with the existing proposals.

INDEX TERMS Mobile cloud computing, searchable encryption, personalized search, updates.

I. INTRODUCTION

Mobile cloud computing [1]–[4] is a new and fundamental model of cloud computing [5], which provides scalable and virtualized storage and computing resources as a service to mobile devices [6], [7]. Nowadays, many companies or data owners are enjoying the convenience of mobile cloud computing by outsourcing their data to cloud servers and later accessing to the data via their mobile devices anywhere anytime. However, noting that the outsourced data may contain sensitive information, e.g. personal emails/videos/photos and financial transactions, privacy issues have long been the major concern and barrier of data outsourcing [8], [9]. For instance, in 2014, hundreds of personal photos of celebrities are stolen from their iCloud. To protect the data privacy [10]–[12], data owners typically encrypt their data before outsourcing to cloud. This, however, weakens the data usability due to the difficulties of searching over the encrypted data. For example, if one needs to search for a relevant document in an encrypted data set, it is unreasonable to first download the whole data set and decrypt it. The situation could be even worse in the mobile environment due to the physical limitations of mobile devices, such as low bandwidth and limited storage capacity.

The searchable encryption [13] represents a promising technique to address the issue by enabling direct search over encrypted data. A number of research works have been focused on the searchable symmetric encryption (SSE) [14], which adopts the same symmetric key to compute the searchable index and trapdoor (encrypted search request). Wang *et al.* [15] propose a vector-space-based search scheme that supports multi-keyword search and result ranking. However, due to the complexity of the key management in its setting, SSE is not suitable to support the multi-data-owner scenario and search authorization that enforces the access policy for each document to only return the search results to the authorized mobile users. This motivates searchable public-key encryption (SPE), which integrates keyword search and public-key encryption. SPE can achieve functionalities in terms of access control and search authorization. For example, Sun *et al.* [11] and Zheng *et al.* [16] leverage attribute-based encryption [17] to propose the attribute-based search scheme with fine-grained access control. However, both [11] and [16] only support single-keyword search and simple conjunctive keyword search. Moreover, SPE is inefficient in practice since it involves many time-consuming asymmetric cryptography operations (e.g., pairing and exponentiation). Therefore, the dynamic property of the searchable encryption including index and document updating remains a challenging issue for both SSE and SPE.

In this paper, we focus on the dynamic searchable encryption in the mobile cloud environment. By exploring the features of mobile cloud computing, we develop a Personalized Search scheme over encrypted data with efficient and secure Updates (PSU), which takes advantages of both SSE and SPE techniques. The main contributions of our work are summarized as follows.

- Efficient and versatile scheme: by leveraging Bloom filter [18] and secure k-nearest neighbor [19] technique, we develop a versatile search scheme that supports multi-keyword search and relevance-based result ranking. By introducing the keyword preferences for mobile users and the score cleaning algorithm on the server side, the proposed scheme could significantly enhance the search accuracy and improve the user search experience accordingly.
- Practical and secured scheme: we modify the attribute-based keyword search [16], [17] and combine it with the vector-space-based search technique in the proposed PSU scheme to achieve the search authorization for the multi-data-owner scenario while maintaining the high search efficiency. Moreover, we adopt the Third-party Auditor (TPA) in the PSU to achieve a highest security level compared with the existing proposals. Via thorough security analysis, we demonstrate the security of PSU in terms of confidentiality, trapdoor unlinkability, forward and backward security, and collusion resistance and hidden access policy.
- Real-world experiment: we run the real-world experiments on a PC server and smart phones to validate the performance of our proposal. It is shown that PSU is more efficient compared with the existing related works in terms of the functionalities, computation and communication overhead.

The remainder of this paper is organized as follows. In Section II, we give an overview of this paper including the system model, security requirements and design goal. In Section III, we recap the Bloom filter, secure kNN technique, and attribute-based encryption with tree-based access structure. In Section IV, we propose the PSU scheme. Its security analysis and performance evaluation are presented in Section V and Section VI, respectively. In Section VII, we present related work. Finally, we close this paper with concluding remarks in Section VIII.

II. PROBLEM FORMULATION

A. SYSTEM MODEL

We consider a system consisting of five components: certificate authority (CA), cloud server (CS), third-party-auditor (TPA), multiple data owners and multiple mobile users. The CA is a globally trusted certificate authority, which takes the responsibility of system setup, mobile user authentication and generation of attribute keys. The TPA is also assumed to be trusted which is responsible for the auditing task in the PSU, including verification of the mobile user's identity and the behavior of the cloud server. It is reasonable since both CA and TPA would be audited by the government office.

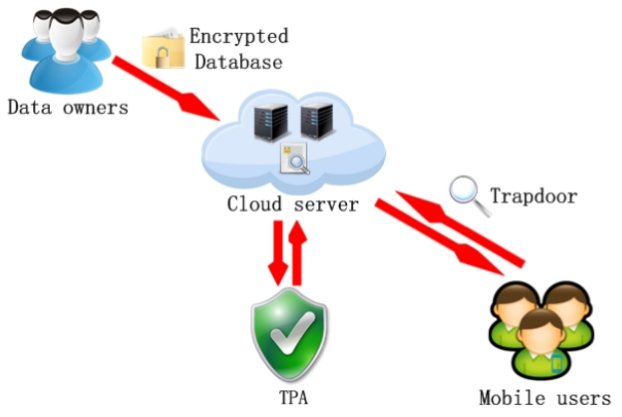


FIGURE 1. **System model.**

The system operates in the following steps. As shown in Fig. 1, the data owners keep a large collection of documents. For each document, they would choose multiple keywords and insert the keywords into the per-document index constructed as a Bloom filter. The data owners then encrypt the per-document index before outsourcing. Since there would be multiple data owners and mobile users in considered system (a data owner could also be a mobile user in some situations), it is therefore necessary to enforce access policy over different documents, and ensure that only the authorized mobile users could access to the documents. Thus, the data owners define the access policy and compute the authorization ciphertext for each document. Finally, the data owners send the encrypted index with the authorization ciphertexts to the cloud server.

When a mobile user intends to search over the encrypted data set, she chooses a keyword set of interest and inserts them into a Bloom filter with different preference weights.

Then, the mobile user computes the search token using her attribute keys and encrypts the query Bloom filter. Upon receiving the search token and the query Bloom filter from the mobile user, the cloud server then calculates the inner products of the index and query Bloom filter, and run the score cleaning algorithm to obtain the final relevance score. After that, the cloud server determines whether the mobile user's attributes satisfy the access policy of the relevant documents ordered by the relevance scores using the search token and the authorization ciphertext. TPA would check the validity of the search results. Finally, the mobile user could obtain the authorized search results.

B. SECURITY REQUIREMENTS

In PSU, Cloud server is honest but assumed to be curious [20]; the Cloud server would execute the assigned task correctly but it is curious on the documents in its storage. Mobile users are dishonest and may collude with each other. Therefore, PSU aims to cover the following security requirements:

- *Confidentiality:* Since the documents may contain sensitive information, they should be kept secret and only authorized mobile users can access to them. Moreover, the cloud server cannot pry into the encrypted index while performing search operations.
- *Trapdoor Unlinkability:* Although the trapdoors are encrypted before being submitted to the cloud server, they can still leak privacy of the mobile users when the cloud server determines the associations between them. The trapdoor unlinkability means that the trapdoors should not be linkable. Specifically, the trapdoors even with the same set of keywords should be totally different, which requires the generation function to be randomized. And the search tokens should also be secret and unlinkable.
- *Forward and Backward Security:* Forward security means that the cloud server does not know that the lately added document contains the keyword searched before. And backward security means that the deleted documents cannot be searched anymore. To support secure dynamic updating, these two security properties must be covered.
- *Collusion Resistance and Hidden Access Control Policy:* The mobile users may intent to combine their search tokens to access to the documents that they cannot access individually. Thus, the proposed scheme should be collusion-resistant. Moreover, the access control policy should not be revealed to the cloud server for privacy considerations.

C. DESIGN GOAL

We target to achieve the following goals in the proposed PSU scheme.

- *Enhancing Search Experience:* The proposed PSU scheme should support personalized multi-keyword search and relevance-based result ranking, since the mobile user would prefer the results of high search accuracy.
- *Privacy Guarantee:* The proposed PSU scheme should cover all the security requirements as listed in the previous subsection.
- *Efficient and Secure Dynamic Updating:* The proposed scheme should support various dynamic updating operations, including the attribute revocation, index and document updates. Moreover, the updates should be both efficient and secure.

III. PRELIMINARIES

A. BLOOM FILTER

A Bloom filter [18] can be used to securely compute the intersection of item sets. Specifically, a (m, k) Bloom filter is an m-dimensional array. Given a data set $S = \{a_1, a_2 \dots a_l\}$ with l elements and k independent universal hash functions $H = \{h_1, h_2 \dots h_k\}$ where $h_i : S \to [0 \dots m-1]$. For each $a_i \in S$, compute $h_j(a_i)$ and set the $h_j(a_i)_{th}$ item in the Bloom filter to 1, where $1 \le j \le k$. To check whether an element a is in the set S or not, compute $h_j(a)$ for each hash functions and check whether the corresponding positions in the Bloom filter is 1. If not, the element is certainly not in S. Otherwise, the element is likely in S with the false positive rate $(1-(1-\frac{1}{m})^{kl})^k$. And when $k = \frac{m}{l}(ln2)$, the Bloom filter achieves an optimal false positive rate $(\frac{1}{2})^k$.

B. SECURE k-NEAREST NEIGHBORS COMPUTATION

We adopt the work of Wong *et al.* [19] in the PSU. Wong et al. propose a secure k-nearest neighbor (kNN) scheme which can confidentially encrypt two vectors and compute Euclidean distance of them. First, the secret key (S, M_1, M_2) should be generated. The binary vector S is a splitting indicator to split plaintext vector into two random vectors. And M_1 and M_2 are used to encrypt the split vectors. The correctness and security of secure kNN computation scheme can be referred to [19]. In the PSU, to securely compute the inner products of the index and the query vector, we modify kNN technique and construct three associated algorithms:

- $Enc_i(p)$. Given an index vector p, this algorithm generates an encrypted query vector P. Specifically, split p into two vectors p' and p'' using the key S as $p'_j = p''_j = p_j,\ if\ S_j = 1;\ p'_j = \frac{1}{2}p_j + r,\ p''_j = \frac{1}{2}p_j - r,\ otherwise$, where r is a random number and p_j is the j-th item of the vector p. Then, compute $P = \{M_1^T \cdot p', M_2^T \cdot p''\}$ as the encrypted index vector.
- $Enc_q(q)$. Given a query vector q, this algorithm generates an encrypted query vector Q. Specifically, split the vector q into two vectors q' and q'' using the key S as $q'_j = q''_j = q_j,\ if\ S_j = 0;\ q'_j = \frac{1}{2}q_j + r',\ q''_j = \frac{1}{2}q_j - r',\ otherwise$, where r' is a random number and q_j is the j-th item of the vector q. Then, compute the $Q = \{M_1^{-1} \cdot q', M_2^{-1} \cdot q''\}$ as the encrypted query vector.

- *Comp*(P, Q). Given the encrypted index and query vector P and Q, this algorithm computes the inner products of the original index and query vector p and q. Specifically, compute the *Score* as follows:

$$\begin{aligned} Score &= P \cdot Q \\ &= \{M_1^T \cdot p', M_2^T \cdot p''\} \cdot \{M_1^{-1} \cdot q', M_2^{-1} \cdot q''\} \\ &= p' \cdot q' + p'' \cdot q'' \\ &= p \cdot q. \end{aligned} \tag{1}$$

C. ATTRIBUTE-BASED ENCRYPTION WITH TREE-BASED ACCESS CONTROL POLICY

Attribute-based encryption (ABE) [17], [21] is a promising technique that can provide fine-grained access control. In the PSU, we adopt CP-ABE technique [17] to achieve search authorization over the outsourced encrypted documents. In CP-ABE, the ciphertexts are created with an pre-defined access policy (e.g., an access tree) and only the mobile users with attributes that satisfy the access policy can decrypt it. Specifically, access policy can be described as a tree structure T in the PSU. Each leaf node of T is associated with an attribute. Each non-leaf node x acts as a threshold gate. Denote the number of the children of node x as n_x and the threshold value as k_x. We can see that $1 \leq k_x \leq n_x$. Node x represents AND gate when $k_x = n_x$ and OR gate when $k_x = 1$. We also denote the associated attribute of leaf node x as $att(x)$. And $p(x)$ denotes the parent of node x. Moreover, we denote $ind(x)$ as the index number of the node x, where $1 \leq ind(x) \leq n_{p(x)}$. Let T_x denote the subtree rooted at node x.

Given a set of attributes Atr and an access tree T, we define the verification function $Verify(Atr, T, x)$ for each node x of T. Specifically, if x is a leaf node, $Verify(Atr, T, x) = 1$ when $att(x) \in Atr$; if x is a non-leaf node, $Verify(Atr, T, x) = 1$ when at least k_x of its children returns 1. The attribute set Atr satisfies the access policy represented by T when $Verify(Atr, T, r) = 1$, where r is the root node. Moreover, an access tree can be used to divide a secret s to a set of values corresponding to each of its leaf node denoted as algorithm $SecretShare(T, s)$ as follows.

SecretShare (T,s): Given a secret s and an access tree T, this algorithm generates a set of secret shares $\{q_x(0)\}$ for each leaf node x. Started with the root node, choose a polynomial q_x for each node x with degree $k_x - 1$. For the root node r, set $q_r(0) = s$ and randomly pick $k_x - 1$ coefficients to fully define the polynomial. For other nodes x, set $q_x(0) = q_{p(x)}(ind(x))$. When this algorithm halts, each leaf node is associated with a secret share $q_x(0)$. As we can see, with at least k_x values of the child node of the node x, one can define the polynomial at node x. That is, given a set of secret shares, one can recover the secret s in a bottom-up manner, which would be discussed in details in our proposed scheme.

IV. PROPOSED SCHEME

In this section, we present the proposed PSU(personalized search) scheme with efficient and secure updates. In PSU, we first adopt the Bloom filter [18] to construct the query and index vector and the kNN computation [19] to securely compute the inner products. We then apply the Third Party Auditor (TPA) and the CP-ABE technique [16], [17], in the PSU to achieve search authorization and forward and backward security. By carefully choosing the different weights in the query and the score cleaning on the server side, PSU not only achieves the personalized search but also significantly increases the search accuracy for the mobile user. PSU proceeds in seven phases, *System Setup*, *Building Encrypted Database*, *Trapdoor Generation*, *Search*, *Auditing*, *Decryption* and, *Efficient and Secure Updating*, which will be described in details below.

A. SYSTEM SETUP

1) CA SETUP

CA is a globally trusted authority, and is in charge of mobile user registration and generating attribute keys. First, CA takes into a security parameter l and outputs two cyclic groups G, G_T of a l-bit prime order p, a generator g of G, and a map $e : G \times G \rightarrow G_T$, such that $e(P_1^a, Q_1^b) = e(P_1, Q_1)^{ab} \in \mathbb{G}_T$ for all $a, b \in \mathbb{Z}_p$ and any $P_1, Q_1 \in \mathbb{G}$. Then, CA chooses k independent universal hash functions $H_B = \{h_1, h_2 \dots h_k\}$ $(h_i : \{0, 1\}^* \rightarrow [0, m-1])$, for a (m, k) Bloom filter. CA also selects a collusion-resistant hash function $H : \{0, 1\}^* \rightarrow G$ modeled as random oracle. CA chooses five random numbers $\alpha, \beta, a, b, c \in Z_p$ and keeps them secret. Moreover, CA would choose a set of n positive numbers $S = \{s_1, s_2, \dots s_n\}$, where $s_i > k \sum_{j=1}^{i-1} s_j$, *for* $2 \leq i \leq n$, where k denotes the number hash functions in the Bloom filter. Finally, CA computes and publish the public parameters Pub as follows.

$$Pub = \{e, g, p, m, k, H_B, H, S, G, G_T, g^a, g^b, g^c, g^\beta, e(g, g)^\alpha\}. \tag{2}$$

2) KEY GENERATION FOR kNN

In the PSU, both the query and the index are built in vector space model. Thus, the data owner first takes a security parameter m (regarding to a m-bit Bloom filter), and outputs the secret key for kNN computations (S, M_1, M_2), where S is a m-dimensional binary vector and M_1, M_2 are two $m * m$ invertible matrixes.

3) MOBILE USER REGISTRATION

CA would authenticate the mobile user's identification and then assign a set of attribute keys according to her role in the system. Attributes of a mobile user would involve many aspects, such as position, age, sex, department, name, and so on. And for each of these attributes, it would include two parts: attribute name and attribute value. That is, for attribute "sex", the specific values could be "male" of "female". For a mobile user with an attribute set Atr (e.g., John Snow, Male, Minister), CA first randomly chooses $r \in Z_p$, a version number $v_j \in Z_p$, and computes $A = g^{(ac-r)/b}$, $B = g^{(\alpha+r)/\beta}$. CA would send the version number to TPA through a secure channel. Then, for each attribute $a_j \in Atr$, CA randomly selects $r_j \in Z_p$ and computes $S_j = (g^r H(a_j)^{r_j})^{v_j}$, $K_j = (g^{r_j})^{v_j}$. The attribute key for the mobile user is $\{Atr, A, B, \{(S_j, K_j)|a_j \in Atr\}\}$ Finally, mobile user would obtain the kNN key from the data owner and the attribute key from CA.

B. BUILDING ENCRYPTED DATABASE

1) ENFORCING ACCESS CONTROL POLICY OVER DOCUMENTS

In the PSU, each data owner keeps a large collection of documents, which should be encrypted before being outsourcing to the cloud server. The data owner firstly chooses a secure symmetric encryption method (e.g., AES-192). And for each document, the data owner would choose a different key κ for the symmetric encryption method. The encrypted form of the document can be denoted as $Enc_\kappa(d)$. Then, this symmetric key could be encrypted using CP-ABE technique [17], where the data owner could enforce the access control policy.

As shown in Section III, the data owner could define the access policy as an access tree. In the PSU, we distinguish the attribute value from the attribute name and only reveal the attribute name to the cloud server. That is, for each leaf node of the access tree, the cloud server only knows the attribute name (e.g., sex) rather than a specific value (e.g., male of female). Specifically, for each document, the data owner defines the access tree T consisting of a set of attribute Ts and chooses a secret $r_2 \in Z_p$. Then, by running the algorithm $SecretShare(T, r_2)$, the data owner would obtain secret share values $q_x(0)$ for each leaf node x. Then, the data owner chooses a version number v_d and computes $W_x = g^{q_x(0)v_d}$ and $D_x = H(att(x))^{q_x(0)v_d}$ for each leaf node x, where $att(x)$ denotes the associated attribute value of the node. Then, the data owner selects two random numbers $r_1 \in Z_p$ to compute $W = g^{cr_1}$, $W_0 = g^{a(r_1+r_2)}g^{br_1}$, $W' = g^{br_2}$, $C = \kappa e(g,g)^{\alpha r_2}$ and $C_1 = g^{\beta r_2}$. That is, the authorization ciphertext Cph of the document is

$$Cph = \{W, W_0, W', C, C_1, \{(W_x, D_x)|att(x) \in Ts\}\}. \quad (3)$$

2) ENCRYPTED INDEX CONSTRUCTION

The index for a document is a (m, k) Bloom filter p, indicating the keywords contained in it. Each item of the p is set to 0 at first. For each document, the data owner would select some keywords. And for a keyword ω, the data owner computes the $h_i(\omega)$ using k different hash functions from H_B and sets the value of the associated position in p as 1. Then, the data owner randomly sets one position in p to 1. The data owner runs the $P = Enc_i(p)$ algorithm to encrypt the Bloom filter. Note that, by introducing the Bloom filter, the PSU gets rid of the limits of the pre-defined keyword dictionary presented in [14] and could achieve a high security level since the meaning of each item in the Bloom filter is not determined.

Finally, the data owner sends the Cph and P to the cloud server.

C. TRAPDOOR GENERATION

To search over the encrypted data set and obtain the authorized results, the mobile user needs to compute the trapdoor including a search token and an encrypted query vector.

1) TOKEN GENERATION

The mobile user chooses $s \in Z_p$ to compute $tk_1 = (g^a g^b)^s$, $tk_2 = g^{cs}$ and $tk_3 = A^s$. Then, for each $a_j \in Atr$, where Atr represents the attribute set of her, the mobile user computes $S'_j = S_j^s$ and $K'_j = K_j^s$. The search token tk of the mobile user is as

$$tk = (tk_1, tk_2, tk_3, \{(S'_j, K'_j)|a_j \in Atr\}). \quad (4)$$

2) QUERY GENERATION

The mobile user takes a keyword conjunction $\varpi = (\omega_1, \omega_2, \cdots \omega_l)$ with l keywords of interest. For each keyword, the mobile user could specify a weight to express her preference for different keywords. Specifically, she would choose l different preference numbers $\{s_1, s_2 \ldots s_l\}$ in an increasing order from the pre-defined set S. Each keyword would be given a preference number and the keyword with more favor would be given a larger one. Then, the mobile user initializes each item of an (m, k) Bloom filter q as 0. For each keyword (e.g., $\omega_i \in \varpi$), the mobile user computes $h_j(\omega_i)$ using the k different hash functions from H_B and increases the value of the associated item in q by s_i. That is, if two hash functions for different keywords collide, the value would be the sum of the preference numbers, which is a little different from the index construction where the item of the index vector is at least set to 1. This preference-based query generation could not only enhance mobile users search experiences but also increase the search accuracy which would be discussed next. The mobile user runs the algorithm $Q = Enc_q(q)$ to compute the encrypted query vector Q.

Finally, the mobile user send the search token tk and the encrypted query vector Q to the cloud server.

D. SEARCH

Upon receiving the trapdoor including the query Q and the search token tk, the cloud server first uses the Q to match with the index P of each document in its storage. By running the algorithm $Score = Comp(P, Q) = p \cdot q$, the cloud server would obtain the relevance score for all the documents. Instead of directly using the inner products to determine the relevancy of the documents as presented in [15], the PSU introduces the keyword preferences and designs a score cleaning algorithm accordingly to significantly increase the search accuracy. Specifically, given $Score$ and the pre-defined set $S = \{s_1, s_2, \ldots s_n\}$, where $s_i > k\sum_{j=1}^{i-1} s_j,\ for\ 2 \le i \le n$, the cloud server runs the Algorithm 1.

After the score cleaning, the cloud server would obtain the more accurate relevance scores of the documents. Then, the cloud server checks whether the mobile user could access to these documents using the authorization ciphertext Cph and

Algorithm 1 Score Cleaning

```
1: Input: $S = \{s_1, s_2, \ldots s_n\}$ and $Score$
   Output: $Score'$
2: Set $T_n = Score$ and $Score' = 0$
3: for $j = n$ to 2 do
4:   if $T_j \geq ks_j$ then
5:     $T_{j-1} = T_j \ mod \ ks_j$
6:     $Score' = Score' + ks_j$
7:   else
8:     while $T_j \geq s_j$ do
9:       $T_j = T_j - s_j$
10:    end while
11:  end if
12: end for
13: if $T_1 == ks_1$ then
14:   $Score' = Score' + T_1$
15: end if
16: return $Score'$
```

search token. As discussed before, the *Cph* only reveals the attribute name to the cloud server. Accordingly, the search token would only also indicate the attribute name to the cloud server. Thus, the cloud server first checks if there would be a potential subset of the mobile user's attributes that satisfies the access control policy defined in the *Cph* of these retrieved documents. If not, the mobile user cannot access to the document. If yes, the cloud server would check the specific attribute values of the mobile user. For each leaf node x and the search token S'_j, K'_j with the same attribute name, the cloud server computes

$$\begin{aligned} E_x &= e(S'_j, W_x)/e(K'_j, D_x) \\ &= \frac{e((g^r H(a_j)^{r_j})^{v_j s}, g^{q_x(0)v_d})}{e((g^{r_j})^{v_j s}, H(att(x))^{q_x(0)v_d})} \\ &= e(g,g)^{rsv_jv_dq_x(0)} \end{aligned} \tag{5}$$

Then, for the non-leaf node x' with threshold value $k_{x'}$, it first computes E_z for all its child nodes $z \in C_{x'}$, where $C_{x'}$ denotes the set of child nodes of x'. If the number of E_z is less than $k_{x'}$, the algorithm halts and return $\perp$. Otherwise, the cloud server combines the values E_z to recover $E_{x'}$ as follows.

$$\begin{aligned} E_{x'} &= \prod_{z \in C_{x'}} E_z^{\Delta_{d,C'_{x'}}(0)}, \quad where \begin{cases} d = ind(z) \\ C'_{x'} = \{ind(z) : z \in C_{x'}\} \end{cases} \\ &= \prod_{z \in C_{x'}} (e(g,g)^{rsv_jv_dq_z(0)})^{\Delta_{d,C'_{x'}}(0)} \\ &= \prod_{z \in C_{x'}} (e(g,g)^{rsv_jv_dq_{p(z)}(d)})^{\Delta_{d,C'_{x'}}(0)} \\ &= \prod_{z \in C_{x'}} (e(g,g)^{rsv_jv_dq_{q(x')}(d)})^{\Delta_{d,C'_{x'}}(0)} \\ &= e(g,g)^{rsv_jv_dq_{x'}(0)} \ (using\ polynomial\ interpolation) \end{aligned} \tag{6}$$

$ind(z)$ represents the index number of node z. The polynomial interpolation recovers the parent node's value by defining the coefficients of the polynomial and computing the $q_x(0)$. For more details, we direct the readers to [17]. When the above algorithm halts at the root node r of the access tree, the cloud server recovers the $E_r = e(g,g)^{rsv_jv_dr_2}$.

Finally, it would send the values E_r and the part of the search token tk_1, tk_2, tk_3 to TPA, and ask TPA to check the validity of the documents and search token. The accuracy and efficiency of the search phase would be discussed in details in Section VI.

E. AUDITING

Upon receiving the E_r and search token from the cloud server, TPA checks if

$$\begin{aligned} &e(W, tk_1)(E_r)^{1/(v_jv_d)} e(tk_3, W') \\ &= e(g^{cr_1}, (g^a g^b)^s) e(g,g)^{rsr_2} e(g^{(acs-rs)/b}, g^{br_2}) \\ &= e(g,g)^{acs(r_1+r_2)+bcsr_1} \\ &= e(g^{a(r_1+r_2)} g^{br_1}, g^{cs}) \\ &= e(W_0, tk_2) \end{aligned} \tag{7}$$

Equation 7 only holds when the attribute keys of the mobile user and the ciphertext of the document are valid and the attribute value of the mobile user satisfies the access control policy embedded in the access tree. Then, TPA would send the most relevant and authorized documents, the associated ciphertext C, C_1 and $E'_r = E_r^{1/(v_jv_d)}$ of these documents to the mobile user.

F. DECRYPTION

Upon receiving the encrypted documents and the associated C, C_1, E'_r, the mobile user computes

$$\begin{aligned} &\frac{C}{e(C_1, B)/(E'_r)^{1/s}} \\ &= \frac{\kappa e(g,g)^{\alpha r_2}}{e(g^{\beta r_2}, g^{(\alpha+r)/\beta}) e(g,g)^{-rr_2}} \\ &= \kappa \end{aligned} \tag{8}$$

Then, the mobile user could use the key κ to further decrypt corresponding encrypted documents.

G. EFFICIENT AND SECURE UPDATING

The PSU should support efficient and secure updating operations, including attribute revocation, index updating and document updating.

1) ATTRIBUTE REVOCATION

In the system, a mobile user's role may dynamically change. Thus, the set of her attributes may need to be altered accordingly or the mobile user should be revoked from the system. In that case, CA needs to re-assign the attribute keys to the mobile user. First, CA regenerates a new version number v'_j for the mobile user and sends the (j, v'_j) to TPA, where j is the id of the mobile user. Then, CA chooses a new random number r' for the mobile user and computes $A'^{(ac-r')/b}, B'^{(\alpha+r')/\beta}$. CA assigns a new set of attributes Atr' to the mobile user. And for each attribute $a_{j'} \in Atr'$, CA randomly selects $r'_j \in Z_p$ and computes $S_{j'} = (g^r H(a_j)^{r'_j})^{v'_j}$, $K_{j'} = (g^{r'_j})^{v'_j}$. Finally, CA sends the new attribute keys $\{Atr', A', B', \{(S_{j'}, K_{j'}) | a_{j'} \in Atr'\}\}$ to the mobile user and the previous attribute keys are invalid since they contain the old version number v_j. In the *Auditing* phase, TPA would check the validity of the mobile user's attribute keys using the new version number v'_j. And the mobile user would not succeed in auditing if she manages to submit the search tokens generated by the outdated attribute keys.

2) INDEX UPDATING

The existing scheme [14] that adopts the vector space model to construct the query and index vector, which requires a pre-defined global dictionary of keywords. This pre-defined dictionary could result in the limitation of dynamic updating of the index, since the keywords of the system are determined at the initial phase of the system. And it would bring much additional computation and storage cost especially when inserting a keyword that is not in the dictionary to the index.

In the PSU, to overcome the limitation, we adopt the Bloom filter technique. When the index of a document needs to be changed or some new keyword ω is introduced to the system, the data owner only needs to compute the $h_i(\omega)$ using k different hash functions and set the associated item in the index p' to 1. Then, the data owner runs the $P' = Enc_i(p')$ algorithm to compute the encrypted index vector P'. This enables a much more flexible and efficient way for the index updating.

3) DOCUMENT UPDATING

The data owners need dynamically add or delete documents on the cloud server. Meanwhile, the data owner would also need to change the access control policy of the outsourced documents sometimes. Moreover, the data owner needs to ensure that the mobile user cannot obtain the documents that are deleted or access to the documents with old access control policy, i.e., enabling the backward security. In the PSU, this could be easily achieved via the auditing of TPA.

When deleting one document, the data owner asks the cloud server to delete it in its storage and then sends the document id to TPA. TPA would add the id to the deletion list. If the cloud server sends the deleted documents and its authorization ciphertext to TPA, TPA would check the deletion list and thus not do the decryption work for the mobile user. And when adding a new document or changing the access control policy of the exited document, the data owner would choose a new symmetric key κ' to encrypt the document. Then, the data owner defines the new access control policy and builds the access tree accordingly. The data owner chooses new random numbers r'_1 and r'_2, as well as a new document version number v'_d and computes the authorization ciphertext as shown in Section IV-B1. Finally, the data owner sends the encrypted document to the cloud server with the new authorization ciphertext and the new document version number to TPA.

V. SECURITY ANALYSIS

Within the system model presented in Section II, this section provides the detailed security analysis on PSU based on the security requirements as listed in Section II.

A. CONFIDENTIALITY

Since there would be multiple data owners in the PSU, confidentiality should be well preserved by enforcing access control policy over the documents. In the PSU, data owners would first choose a symmetric key to encrypt the documents. Specifically, the data owners could choose different keys for each document or for a set of documents. Then, for the documents with the same symmetric key, data owners would define the access control policy and encrypt the symmetric key using ABE technique. That is, only mobile users with attribute keys that satisfy the access control policy of the documents could obtain the symmetric key and further decrypt the document. Moreover, the per-document index is encrypted using the kNN key, the cloud server could not deduce any additional information while performing search operations. In addition, adopting Bloom filter to construct the index rather than using a pre-defined dictionary would make the index more indistinguishable for more the cloud server. Thus, the confidentiality of documents and index is well preserved in the PSU.

B. TRAPDOOR UNLINKABILITY

In the PSU, trapdoor consists of the encrypted query vector and search token. When generating the query vector represented as a (m, k) Bloom filter, the mobile user would compute k hash values using the k different hash functions and set the corresponding position in the Bloom filter to a specific preference number. Moreover, the mobile user could set some dummy positions to the preference number while not affecting the search accuracy. Thus, the two query vector containing the same set of keywords could not be the same for the different choices of the preference weights and dummy positions. Then, the query vectors are encrypted using the kNN key (S, M_1, M_2), which makes the encrypted query vector totally different. As for the search token, CA would choose a random number r to generate the attribute keys for each mobile user, which makes the attribute keys with the same attribute for different mobile users unlinkable. Moreover, the mobile user would encrypt her attribute keys by randomly choosing a secret s when generating the search token. That is, search tokens for the same mobile user would be totally different and unlinkable. Thus, the PSU could achieve the trapdoor unlinkability.

C. FORWARD AND BACKWARD SECURITY

The PSU supports dynamic operations while preserving the forward and backward security. As for the forward security, the newly added document with the per-document index is encrypted before being outsourced to the cloud server. As mentioned before, the data owner and the mobile user could add some dummy numbers in the index or query vector, which makes the encrypted form totally different. That is, the generation functions for both index and query vector are randomized rather than determined. Thus, the cloud server could not deduce the specific keyword information in the encrypted index and query vector and the forward security is ensured.

On addressing the backward security problem, we adopt the third-party auditor technique in the PSU. When computing the index for the document, the data owner would choose a secret document version number v_d for the document and send it to TPA. If the content of the document or the access control policy is changed, the data owner needs to re-compute the authorization ciphertext with a new version number v'_d. Meanwhile, when generating the attribute keys for the mobile user, CA would also choose a secret user version number v_j and send it to TPA. When some of the mobile user's attributes need to be changed, it requires CA to select a new user version number v'_j to generate the new set of attribute keys. And TPA is responsible to check the behaviors of the mobile user and server during the Auditing phase. When the cloud server returns the deleted document or the document with outdated access control policy to the mobile user, the equation 7 would not hold since the document does not contain the new version number v'_d, which goes the same for the situation when the mobile user tries to obtain the document with outdated attribute keys. That is, the backward security are well protected in the PSU.

D. COLLUSION RESISTANCE AND HIDDEN ACCESS CONTROL POLICY

The mobile users are assumed to be dishonest in the PSU and may intend to combine their search tokens to access to the documents that they cannot access to individually. However, when generating attribute keys, CA would choose two random numbers r and v_j for the mobile user. Thus, if two mobile users tries to combine their search tokens, the equation 7 would not establish in the Auditing phase. And existing schemes, such as [11] and [16], require the data owner to reveal the access control policy to the cloud server, which may lead to the information leakage of the documents. In the PSU, we separate the attribute values from the attribute name. Specifically, the data owner only needs to reveal the attribute name to the cloud server, e.g. "homeland". And the specific value of this attribute may be "New York" or "Florida". Only the search token with correct attribute value could decrypt the authorization ciphertext. This is reasonable since revealing the attribute name would only leak little information to the cloud server. Moreover, the generation function for search token is randomized, which could prevent the cloud server from deducing the access control policy from the search tokens via dictionary attack. That is, the PSU could achieve collusion resistance and partially hidden access policy.

TABLE 1. **Comparison of security level.**

	[16]	[15]	[11]	PSU
Confidentiality	√	√	√	√
Trapdoor Unlinkability	√	√	√	√
Forward Security	√	√	√	√
Backward Security				√
Collusion Resistance	√	N/A	√	√
Hidden Access Policy		N/A		√

As shown in Table 1, we compare the security level among the scheme [11], [15], [16] and the PSU. As we can see, while [15] does not consider the search authorization problem, the PSU achieves the highest security properties.

VI. PERFORMANCE EVALUATION

In this section, we conduct real-world experiments to show the performance of the proposed PSU, and compare the PSU with the existing schemes [11], [15], [16] from the aspects of functionalities, computation and communication overhead, respectively.

TABLE 2. **Comparison of functionalities.**

	[16]	[15]	[11]	PSU
Multi-keyword		√	√	√
Result Ranking		√		√
Search Authorization	√		√	√
Dynamic Updates		√	√	√
Personalized Search				√

A. FUNCTIONALITIES

Table 2 summarizes the functionalities provided by PSU, [11], [15], [16]. PSU supports multi-keyword personalized search and relevance-based result ranking, which would significantly increase the search accuracy. In addition, PSU enables search authorization and dynamic updates, which are important functions for practical deployments. Compared with PSU, most existing works as in Table 2 only support partial functionalities as listed.

B. ACCURACY ANALYSIS

To support multi-keyword ranked search which has not been addressed in [11] and [16], we introduce the vector space model and Bloom filter in the PSU. In Wang's scheme [15], the authors use the inner products of the index and the query vector constructed by Bloom filter as the relevance score. Since the collision arises when the number of keywords inserted to the index increases, the score may be "dirty" (obscure) and lead to a decline of the search accuracy accordingly. For example, when there do not exist common keywords between the per-document index and query, the cloud server may still return the irrelevant documents according to the scores. In specific, the probability that any two different keywords collide for only one-bit in the (m, k) Bloom filter is $P_1 = 1 - (1 - \frac{1}{m})^{kn}$, where n is the number of the keywords in the index. Therefore, P_1 would significantly increase when inserting many keywords to the index. Moreover, since the mobile users typically select a few keywords only for their search requests [22], the relevance scores for the documents could be hard to distinguish [23] in this scenario.

PSU fixes the issue by introducing preference weights for different keywords to construct the Bloom filter, instead of setting the item to 1. PUS is further equipped with a score cleaning algorithm to eliminate the noise caused by the Bloom filter. Therefore, in PSU, the relevance score would be affected if and only if there exist k-collisions in the index and query vector with the probability $P_2 = (1 - (1 - \frac{1}{m})^{kn})^k$. This increases the search accuracy compared with [16]. Meanwhile, the weights of the different keywords would lead to a more personalized search results.

C. COMPUTATION OVERHEAD

In this subsection, we conduct real-world experiments to evaluate the performance of the proposed PSU scheme from the aspects of encrypted database setup, trapdoor generation, search and updating, respectively. Specifically, we apply a real data set from National Science Foundation Research Awards Abstracts 1990-2003 [24], and select some documents and associated keywords. We conduct our experiments on a laptop computer with Core i3 2.13GHz processor and 4 GB memory using C and JAVA code. Moreover, we run the trapdoor generation phase on a real mobile phone (Huawei Honor 6) with Hisilicon Kirin 920 processor and 3 GB memory using JAVA code. For the implementation of the bilinear map, we adopt the Java Pairing-Based Cryptography Library (JPBC) [25]. For the (m, k) Bloom filter, we choose $m = 10000$ and $k = 10$ and implement it using SHA-1 hash function. And we use some notations here. In specific, we denote T_g and T_{gt} as the time for an exponentiation operation in G and G_T, respectively, and T_p as the time for a paring operation. And we denote the n as the number of attributes in a authorization ciphertext and N as the number of attributes in the system.

1) ENCRYPTED DATABASE SETUP

To build the encrypted database, the data owner needs to compute the per-document index and authorization ciphertext. For the index, the data owner selects some keywords and computes k hash values using the hash functions from H_B and sets the associated item in the m-bit Bloom filter p to 1. Then, the data owner encrypts the p using the kNN technique. For each per-document index, the data owner performs two multiplications of a $m * m$ matrix and a m-dimension vector. The computational complexity is $O(m^2)$. Thus, the time for computing all the encrypted index is linear to the number of documents in the data set.

For the computation of the authorization ciphertext Cph, it would take the data owner approximately $(2n+5)T_g + T_{gt}$ time for a single document. Compared with that in Sun's scheme [11], it would take the data owner approximately $(N + 1)T_g + T_{gt}$ time. While time cost is linearly increasing with the number of the attributes in the whole system in Sun's scheme [11], the PSU is much efficient since it is mainly affected by the number of documents in a ciphertext. And we run the real experiments to evaluate the execution time as shown in Fig. 2. For each document, we randomly select about 70 different keywords. And we set the access control policy for each document as a combination of AND gates. As we can see, time cost for building the encrypted index is mainly affected by the number of documents in the data set and time cost for a authorization ciphertext linearly increases with the number of attributes involved. Note that, the data owner only needs to build the encrypted database once and therefore the computation overhead is acceptable for practical uses.

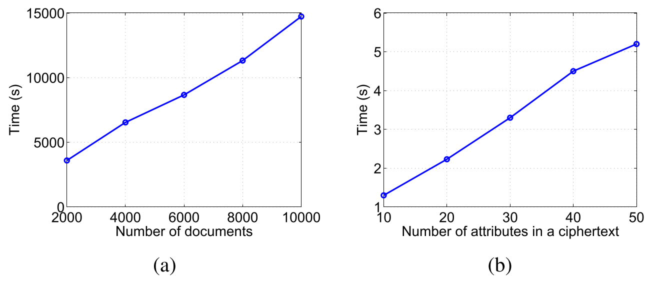
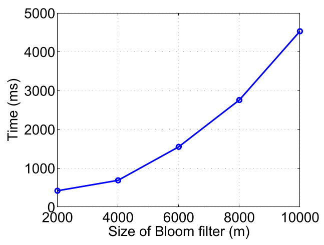


FIGURE 2. Computation overhead for building the encrypted database. (a) Time cost for generating the encrypted index for different number of documents in the system with the same size of Bloom filter, where m = 10000 and k = 10. (b) Time cost for generating a per-document authorization ciphertext for different number of attributes involved.

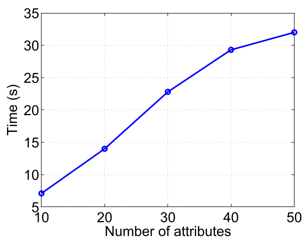
2) TRAPDOOR GENERATION

The trapdoor in the PSU includes the encrypted query vector and the search token. To generate the query vector q, the mobile user takes a keyword set of interest and insets them into the m-bit Bloom filter. Then, the mobile user encrypts the q using kNN technique with the complexity $O(m^2)$. As for the search token, the mobile user needs to choose a subset of her attribute keys to satisfy the access control policy and compute the search token accordingly. It would approximately take the mobile user $(2s + 4)T_g$ time, where s denotes the number of attributes in the attribute subset. Compared with that in Sun's scheme [11], it would approximately take the user $(2N + 1)T_g$, which is lineally increasing with the number of the attributes in the whole system, the PSU is more efficient.

Moreover, we run experiments on a real smartphone. Specifically, the mobile user would select 10 keywords for the search request. As shown in Fig. 3 (a), the execution of time for computing the encrypted query vector is increasing with the size of the Bloom filter when other parameters are set. And the time cost for computing the search token is linearly increasing with the number of attributes involved. As shown in Fig. 3 (b). Note that, a larger size of the Bloom filter

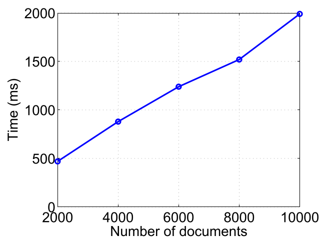


(a)

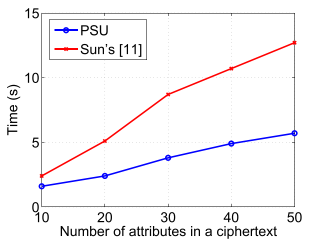


(b)

FIGURE 3. Computation overhead for trapdoor generation on a real smartphone. (a) Time cost for generating the encrypted query vector for different size of Bloom filter with the same of keywords of interest. (b) Time cost for generating a search token for different number of attributes involved.



(a)



(b)

FIGURE 4. Computation overhead for calculating relevance score and verification. (a) Search time for different number of documents. (b) Time cost for verifying one single document with the different number of attributes in a ciphertext.

would significantly decrease the collision rate but increase the computation cost. Thus, there exists a trade-off between efficiency and utility. And the access control policy may not be that complex such that the number of attributes involved could be reduced. That is, overall efficiency of the trapdoor phase could be restricted in an acceptable level.

3) SEARCH

In the attribute-based keyword search scheme [11], [16], searching over one single document would require many time-consuming operations (e.g. paring and exponentiation). In this case, time cost for searching over the whole database could be unacceptable when there is a large number of documents on the cloud server, which makes it difficult for adopting the attribute-based keyword search for practical uses.

In PSU, by leveraging vector space model, the search efficiency could be significantly improved. Specifically, upon receiving the trapdoor, the cloud server first uses the encrypted query vector Q to search over the encrypted index and obtains the documents that are more relevant to the search request. Then, the cloud server could ask TPA to determine whether the mobile user could access to the documents using the authorization ciphertext and search token, rather than performing the attribute-based search over the whole database. Moreover, when the relevance score of the documents is under the specific threshold value, the cloud server stops the authorizing operations since these documents are irrelevant to the mobile user's search request. To summarize, PSU is much more efficient by avoiding unnecessary authorizing operations.

The experimental results are shown to consolidate our claim. For the search operation, we assume that the encrypted index has been loaded in the memory to avoid time-consuming I/O operations. As shown in Fig. 4(a), we can see that although the time for calculating relevance score is linearly increasing with the number of the documents in the system, it is still a quite light-weight process compared with the verification operation. Moreover, given the relevance scores, PSU only requires the cloud server to perform the verification operation over the more relevant documents instead of the whole data set. Then, we compare the time cost for verifying one single document between PSU and Sun's scheme [11] by assuming that the number of attributes in the system is five times of that in a ciphertext. As shown in Fig. 4(b), although the time costs in both schemes are linearly increasing, the increase rate of PSU is less than that in Sun's scheme.

4) UPDATING

By introducing TPA in PSU, the data owner could dynamically update the documents and associated index efficiently and securely. Specifically, the data owner only needs to re-compute the encrypted document and index. Then, the data owner asks TPA and the cloud server to update the corresponding information. Although PSU achieves almost the same computation overhead for document and index updating compared with [11] and [16], PSU obtains a higher security level including forward and backward security as discussed in Section V. For the attribute revocation problem, as it is only considered in Sun's [11], we would only compare PSU with [11].

In [11], when one of the mobile user is revoked from the system, the cloud server needs to update all the authorization ciphertexts that are involved with the revoked attribute and the attribute keys of the non-revoked mobile users who hold the revoked attribute, resulting in $(N_c + N_s)T_g$ time, where N_c and N_s denote the number of these authorization ciphertexts and mobile users, respectively. While in PSU, CA only needs to re-compute a set of attribute keys for the revoked mobile user with execution time $(3s + 2)T_g$, where s is the number of the attributes a mobile user holds.

In the next experiment, we show the efficiency of PSU. We assume that the number of attributes in the system is five times of the number of attributes of a mobile user. As shown in Fig. 5(a), we compare the time cost for the generation of attribute keys between PSU and Sun's scheme [11]. Since time cost for key generation is linearly increasing with the number of attributes of a mobile user in PSU rather than the number of attributes in the system as that in Sun's, we can see that the increase rate of PSU is much less than that of Sun's. Then, we assume that a mobile user could hold at least 20 attributes and compare the time cost for attribute

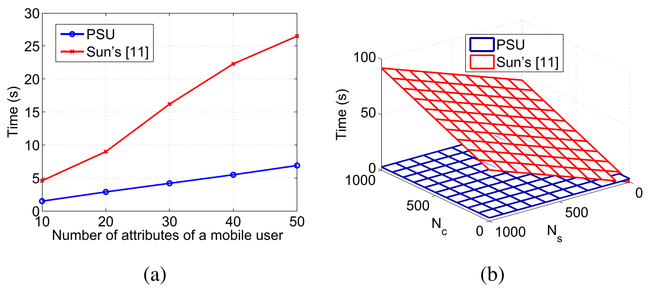


FIGURE 5. **Computation overhead for attribute revocation. (a) Time for generating the attribute keys for the different number of attributes of a mobile user in PSU. (b) Comparison for attribute revocation for the different number of mobile user and authorization ciphertext involved.**

revocation with [11] as shown in Fig. 5(b). We can see that while the execution time is linearly increasing in Sun's scheme [11] with the number of authorization ciphertexts (N_c) and the mobile users (N_s) involved, time cost for attribute revocation is constant in the PSU.

D. COMMUNICATION OVERHEAD

Since the search authorization problem has not been considered in [15], we only compare the communication overhead among PSU, [11], [16]. Once the system is setup, the communication overhead is mainly determined by the search and updates. As for the search operations, PSU supports multi-keyword personalized search and result ranking; [11] and [16] supports single-keyword search or simple conjunctive keyword search. Therefore, with more informative keywords, the mobile user would obtain the results that are more relevant to their search request in PSU. This accordingly can significantly reduce the unnecessary Internet traffic by only sending back the top documents rather than undifferentiated results. As for the attribute revocation, PSU incurs much less communication cost since it only requires to send a new set of attribute keys to the revoked mobile user while it requires to update the associated authorization ciphertexts and attribute keys in Sun's scheme [11].

VII. RELATED WORK

The Searchable Encryption (SE) has recently attracted keen research efforts [26]–[30]. Song *et al.* [31] first propose the concept of the searchable symmetric encryption (SSE), which can support single keyword search. Curtmola *et al.* [32] extends the work by giving the formal definitions of searchable encryption and security model. The basic idea of [32] is to introduce a keyword-based search index. However, the construction of the index and search token is for deterministic encryption and therefore reveals the search keywords to the cloud server with strong background knowledge [14]. Based on [32], subsequent works, such as [33] have been developed to extend the searchable symmetric encryption to a dynamic setting. However, these proposals still require deterministic encryption method to construct the search token.

Li *et al.* [14] adopt the vector space model and design a search scheme that supports multi-keyword search and relevance-based result ranking. However, the proposed scheme [14] requires a pre-defined dictionary for all the keywords in the system, which limits the dynamic properties of the SSE. Moreover, search authorization problem, which is of great significance for multiple-contributors scenario, has not been addressed in [14]. Later, Li *et al.* [34] propose an authorized multi-keyword ranked search scheme that leverages attribute-based encryption and vector space model to provide the search users both good search experience and search authorization. However, the proposed search scheme in [34] still requires a pre-defined dictionary in the system thus lacks in efficiency in dynamic updating. To solve the problem of the pre-defined dictionary, Wang *et al.* [15] propose a multi-keyword search scheme that adopts the Bloom filter [18] instead of a pre-defined dictionary in the proposal, to construct the search index and trapdoor. The proposed search scheme in [15], however, may reduce the accuracy of the search results due to the false positive caused by the Bloom filter.

Besides SSE, Boneh *et al.* [35] first propose the concept of the searchable public-key encryption (SPE) and design a search scheme based on the public key setting that supports single keyword search. Attribute-based encryption (ABE) [17], [36] is a promising technique that could provide fine-grained access control over the outsourced data. Sun *et al.* [11] and Zheng *et al.* [16] leverage ABE with SPE technique to achieve authorized keyword search, which returns the results to the authorized users only. Sun *et al.* [11] propose an attribute-based keyword search scheme with fine-grained owner-enforced search authorization. The proposal also supports user revocation by adopting the proxy re-encryption technique. However, [11] may require performing time-consuming operations over the entire data set and could hardly guarantee the backward security when the documents on the cloud server needs to be updated. Zheng *et al.* [16] propose an attribute-based keyword search over outsourced encrypted data. However, their scheme does not support the dynamic operations.

VIII. CONCLUSION

In this paper, we have proposed a personalized search scheme with efficient and secure updates in mobile clouds. We have conducted extensive security analysis to demonstrate that the proposed scheme could achieve confidentiality, trapdoor unlinkability, forward and backward security, and collusion resistance and hidden access policy. Using extensive experiments, we have shown that our proposed scheme is more efficient in terms of functionalities, computation and communication overhead compared with the existing schemes. For the future work, we intend to further research on the dynamic nature of the searchable encryption technique and test our proposal in the real-world cloud platform.

REFERENCES

[1] Q. Shen, X. Liang, X. Shen, X. Lin, and H. Y. Luo, "Exploiting geo-distributed clouds for a E-health monitoring system with minimum service delay and privacy preservation," *IEEE J. Biomed. Health Informat.*, vol. 18, no. 2, pp. 430–439, Mar. 2014.

[2] H. Liang, L. X. Cai, D. Huang, X. Shen, and D. Peng, "An SMDP-based service model for interdomain resource allocation in mobile cloud networks," *IEEE Trans. Veh. Technol.*, vol. 61, no. 5, pp. 2222–2232, Jun. 2012.

[3] Y. Cai, F. Yu, and S. Bu, "Cloud computing meets mobile wireless communications in next generation cellular networks," *IEEE Netw.*, vol. 28, no. 6, pp. 54–59, Nov./Dec. 2014.

[4] D. Zeng, S. Guo, I. Stojmenovic, and S. Yu, "Stochastic modeling and analysis of opportunistic computing in intermittent mobile cloud," in *Proc. 8th IEEE Conf. Ind. Electron. Appl. (ICIEA)*, Jun. 2013, pp. 1902–1907.

[5] S. Yu, R. Doss, W. Zhou, and S. Guo, "A general cloud firewall framework with dynamic resource allocation," in *Proc. ICC*, Jun. 2013, pp. 1941–1945.

[6] F. R. Yu and V. Leung, *Advances in Mobile Cloud Computing Systems*. Boca Raton, FL, USA: CRC Press, 2015.

[7] Y. Cai, F. R. Yu, and S. Bu, "Dynamic operations of cloud radio access networks (C-RAN) for mobile cloud computing systems," *IEEE Trans. Veh. Technol.*, 2015. DOI: 10.1109/TVT.2015.2411739

[8] K. Zhang, X. Liang, M. Baura, R. Lu, and X. Shen, "PHDA: A priority based health data aggregation with privacy preservation for cloud assisted WBANs," *Inf. Sci.*, vol. 284, pp. 130–141, Nov. 2014.

[9] M. M. E. A. Mahmoud and X. Shen, "A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 10, pp. 1805–1818, Oct. 2012.

[10] W. Ren, "uLeepp: An ultra-lightweight energy-efficient and privacy-protected scheme for pervasive and mobile WBSN-cloud communications," *Ad Hoc Sensor Wireless Netw.*, vol. 27, nos. 3–4, pp. 173–195, 2015.

[11] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," in *Proc. IEEE INFOCOM*, Apr./May 2014, pp. 226–234.

[12] D. M. B. Ying and H. T. Mouftah, "Sink privacy protection with minimum network traffic in WSNs," *Ad Hoc Sensor Wireless Netw.*, vol. 25, nos. 1–2, pp. 69–87, 2015.

[13] H. Li, Y. Yang, H. Yang, and M. Wen, "Achieving efficient and privacy-preserving multi-feature search for mobile sensing," *Comput. Commun.*, vol. 65, pp. 35–42, Jul. 2015.

[14] H. Li, Y. Yang, T. Luan, X. Liang, L. Zhou, and X. Shen, "Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data," *IEEE Trans. Dependable Secure Comput.*, 2015. DOI: 10.1109/TDSC.2015.2406704

[15] B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud," in *Proc. IEEE INFOCOM*, Apr./May 2014, pp. 2112–2120.

[16] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: Verifiable attribute-based keyword search over outsourced encrypted data," in *Proc. INFOCOM*, Apr./May 2014, pp. 522–530.

[17] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, May 2007, pp. 321–334.

[18] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.

[19] W. K. Wong, D. W.-L. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2009, pp. 139–152.

[20] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward cloud-based vehicular networks with efficient resource management," *IEEE Netw.*, vol. 27, no. 5, pp. 48–55, Sep./Oct. 2013.

[21] H. Li, Y. Dai, L. Tian, and H. Yang, "Identity-based authentication for cloud computing," in *Proc. 1st Int. Conf. Cloud Comput.*, 2009, pp. 157–166.

[22] D. Lagun, C.-H. Hsieh, D. Webster, and V. Navalpakkam, "Towards better measurement of attention and satisfaction in mobile search," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2014, pp. 113–122.

[23] J. Zhang, K. Mouratidis, and H. Pang, "Global immutable region computation," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 1151–1162.

[24] *NSF Research Awards Abstracts 1990–2003*. [Online]. Available: http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html, accessed Jun. 2015.

[25] *Java Pairing-Based Cryptography Library (JPBC)*. [Online]. Available: http://gas.dia.unisa.it/projects/jpbc, accessed Jun. 2015.

[26] Y. Yang, H. Li, M. Wen, H. Luo, and R. Lu, "Achieving ranked range query in smart grid auction market," in *Proc. ICC*, Jun. 2014, pp. 951–956.

[27] Y. Yang, H. Li, W. Liu, H. Yao, and M. Wen, "Secure dynamic searchable symmetric encryption with constant document update cost," in *Proc. GLOBECOM*, Dec. 2014, pp. 775–780.

[28] H. Li, Y. Yang, M. Wen, H. Luo, and R. Lu, "EMRQ: An efficient multi-keyword range query scheme in smart grid auction market," *KSII Trans. Internet Inf. Syst.*, vol. 8, no. 11, pp. 3937–3954, 2014.

[29] H. Li, D. Liu, Y. Dai, T. H. Luan, and X. S. Shen, "Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 1, pp. 127–138, Mar. 2015.

[30] H. Li, D. Liu, Y. Dai, and T. H. Luan, "Engineering searchable encryption of mobile cloud networks: When QoE meets QoP," *IEEE Wireless Commun.*, vol. 22, no. 4, pp. 74–80, Aug. 2015.

[31] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Symp. Secur. Privacy*, May 2000, pp. 44–55.

[32] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, 2006, pp. 79–88.

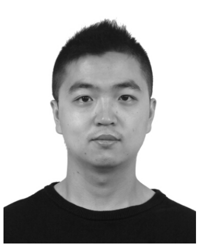
[33] M. Naveed, M. Prabhakaran, and C. A. Gunter, "Dynamic searchable encryption via blind storage," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 639–654.

[34] H. Li, D. Liu, K. Jia, and X. Lin, "Achieving authorized and ranked multi-keyword search over encrypted cloud data," in *Proc. ICC*, London, U.K., Jun. 2015, pp. 7450–7455.

[35] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. EUROCRYPT*, 2004, pp. 506–522.

[36] D. Liu, H. Li, Y. Yang, and H. Yang, "Achieving multi-authority access control with efficient attribute revocation in smart grid," in *Proc. ICC*, Jun. 2014, pp. 634–639.

HONGWEI LI (M'11) received the Ph.D. degree in computer software and theory from the University of Electronic Science and Technology of China, China, in 2008. He was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, for one year. He is an Associate Professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He is also with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing. His research interests include network security, applied cryptography, and trusted computing. He is a member of the China Computer Federation and the China Association for Cryptologic Research. He serves as the Associate Editor of *Peer-to-Peer Networking and Applications* and the Guest Editor of *Peer to-Peer Networking and Applications* of the Special Issue on Security and Privacy of P2P Networks in Emerging Smart City. He also serves on the technical program committees for many international conferences, such as the IEEE INFOCOM, the IEEE ICC, the IEEE GLOBECOM, the IEEE WCNC, the IEEE SmartGridComm, BODYNETS, and the IEEE DASC.



DONGXIAO LIU (S'14) received the B.S. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, China, in 2013, where he is currently pursuing the master's degree. His research interests include cryptography and cloud computing security. He serves as a reviewer of *Peer-to-Peer Networking and Application*.

YUANSHUN DAI (M'03) received the B.S. degree from Tsinghua University, Beijing, China, in 2000, and the Ph.D. degree from the National University of Singapore, Singapore, in 2003. He is the Dean of the School of Computer Science and Engineering with the University of Electronic Science and Technology of China. He is also a Chaired Professor and the Director of the Collaborative Autonomic Computing Laboratory. He has served as the Chairman of the Professor Committee in the School of Computer Science and Engineering since 2012, and the Associate Director at the Youth Committee of the National 1000er Plan in China. He has authored more than 100 papers and five books, where there are 50 papers indexed by SCI, including 25 IEEE/ACM Transactions papers. His current research interests include cloud computing and big data, reliability and security, modeling, and optimization. He has served as a Guest Editor of the IEEE Transactions on Reliability. He is also on the Editorial Boards of several journals.

TOM H. LUAN (M'13) received the B.Sc. degree from Xi'an Jiaotong University, China, in 2004, the M.Phil. degree from the Hong Kong University of Science and Technology, Hong Kong, China, in 2007, and the Ph.D. degree from the University of Waterloo, Canada, in 2012. Since 2013, he has been the Lecturer in Mobile and Applications with the School of Information Technology, Deakin University, Melbourne, Australia. His research mainly focuses on vehicular networking, wireless content distribution, peer-to-peer networking, and mobile cloud computing.

SHUI YU (M'05–SM'12) is currently a Senior Lecturer with the School of Information Technology, Deakin University. He has authored two monographs and edited one book, more than 150 technical papers, including top journals and top conferences, such as the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Computers, the IEEE Transactions on Information Forensics and Security, the IEEE Transactions on Mobile Computing, the IEEE Transactions on Knowledge and Data Engineering, the IEEE Transactions on Emerging Topics in Computing, and the IEEE INFOCOM. His research interests include networking theory, cyber security, mathematical modeling, and big data. He is a member of the Deakin University Academic Board (2015-2016) and AAAS, the Vice Chair of the Technical Subcommittee on Big Data Processing, Analytics, and Networking of the IEEE Communication Society. He initiated the research field of networking for big data in 2014. He has an h-index of 18. He actively serves his research communities in various roles. He serves on the Editorial Boards of the IEEE Transactions on Parallel and Distributed Systems, the IEEE Communications Surveys and Tutorials, the IEEE Access, and a number of other international journals. He has served more than 50 international conferences as a member of the Organizing Committee, such as the Publication Chair of the IEEE Globecom 2015 and the IEEE INFOCOM 2016, the TPC Co-Chair of the IEEE BigDataService 2015 and the IEEE ATNAC 2014 and 2015.