Received December 13, 2018, accepted December 20, 2018, date of publication January 3, 2019,
date of current version January 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2889754

CP-ABSE: A Ciphertext-Policy Attribute-Based Searchable Encryption Scheme

HUI YIN[1], JIXIN ZHANG[2], YINQIAO XIONG[1,3], LU OU[2], (Member, IEEE), FANGMIN LI[1], SHAOLIN LIAO[4,5], (Senior Member, IEEE), AND KEQIN LI[6], (Fellow, IEEE)

[1]College of Computer Engineering and Applied Mathematics, Changsha University, Changsha 410022, China
[2]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[3]College of Computer, National University of Defense Technology, Changsha 410073, China
[4]Argonne National Laboratory, Lemont, IL, 60439, USA
[5]Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL 60616, USA
[6]Department of Computer Science, The State University of New York, New Paltz, NY12561, USA

Corresponding authors: Jixin Zhang (zhangjixin@hnu.edu.cn) and Yinqiao Xiong (yq.xiong@ccsu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772088, in part by the Natural Science Foundation of Hunan Province under Grant 2017JJ2292, in part by the Science and Technology Key Projects of Hunan Province under Grant 2016JC2012, in part by the Outstanding Youth Research Project of Provincial Education Department of Hunan under Grant 17B030, in part by the Science and Technology Planning Project of Changsha under Grant K1705018, Grant ZD1601042, and Grant K1705031.

ABSTRACT Searchable encryption provides an effective mechanism that achieves secure search over encrypted data. A popular application model of searchable encryption is that a data owner stores encrypted data to a server and the server can effectively perform keyword-based search over encrypted data according to a query trapdoor submitted by a data user, where the owner's data and the user's queries are kept secret in the server. Recently, many searchable encryptions have been proposed to achieve better security and performance, provide secure data updatable feature (*dynamics*), and search results verifiable capability (*verifiability*). However, most of the existing works endow the data user an unlimited search capacities and do not consider a data user's search permissions. In practical application, granting search privileges for data users is a very important measure to enforce data access control. In this paper, we propose an attribute-based searchable encryption scheme by leveraging the ciphertext-policy attribute-based encryption technique. Our scheme allows the data owner to conduct a fine-grained search authorization for a data user. The main idea is that a data owner encrypts an index keyword under a specified access policy, if and only if, a data user's attributes satisfy the access policy, the data user can perform search over the encrypted index keyword. We provide the detailed correctness analyses, performance analyses, and security proofs for our scheme. The extensive experiments demonstrate that our proposed scheme outperforms the similar work CP-ABKS proposed by Zheng on many aspects.

INDEX TERMS Access control, attribute-based encryption, search authorization, searchable encryption.

I. INTRODUCTION

A. MOTIVATION

Currently, with the increasing popularity of cloud based outsourcing service paradigm, more and more data have been assembled to a cloud server center. These data may be individual users' social data, business data of enterprises, the collected data from sensors, and so on. At the same time, data outsourcing raises confidentiality and privacy concerns, as the cloud server is not always fully trusted [2]. An effective measure of solving the concerns is to encrypt data before outsourcing them to the cloud server [3]. However, traditional symmetric encryptions and asymmetric encryptions are blind for the search over ciphertext. In such a background, searchable encryption emerges at the right moment.

In summary, searchable encryption is a cryptographic primitive that enables searching on the encrypted data while protecting the confidentiality of data and queries [4]. Similar to the traditional encryption, searchable encryption can also be classified into symmetric searchable encryption and asymmetric searchable encryption. Due to more excellent search performance, since the first searchable encryption scheme was constructed in the symmetric environment, formally named *Searchable Symmetric Encryption* (SSE), SSE has been fully studied. Many advanced SSE schemes have been proposed to pursue for more practical properties such as efficient data updating (*dynamics*) [5], secure search results verification (*verifiability*) [6], forward privacy [7], backward privacy [8], etc. The first public-key based searchable encryption scheme, *Public-Key Encryption with Keyword Search* (PEKS) [9], was proposed to achieve secure email search in the ciphertext environment. To rich search functionality, conjunctive search, subset search, and range search were also explored in the public-key cryptosystem such as [10]–[13].

However, all above mentioned searchable encryption schemes have a common assumption that a search user is endowed unlimited search capability, who can adopt any keyword to obtain encrypted data containing the query keyword from the server. As a result, the data owner cannot enforce effective data access control for outsourced encrypted data. Data access control is a practical requirement in some real systems. For example, a financial executive of a company can search and manage all commodity orders while a sales manager has no such permissions. To enable search and flexible data access control over encrypted data simultaneously, researchers propose to design searchable encryption with keyword search authorization by leveraging the attribute-based encryption technique. Attribute-based encryption (ABE) [14] is a novel cryptographic primitive that can enforce fine-grained data access control via cryptographic means. In ABE, an access policy is an important component, depending on where the access policy is equipped with, ABE has two variants: key-policy ABE (KP-ABE) [15] and ciphertext-policy ABE (CP-ABE) [16]. In KP-ABE, the key is associated to the access policy and the data are encrypted by a set of attributes, a user can decrypt the ciphertext if and only if the access policy in her key satisfies the attribute set in the ciphertext. On the completely contrary to KP-ABE, in CP-ABE, the access policy is embedded into the ciphertext, and a user's key is associated with her attributes. Only when a user's attributes satisfy the access policy in the ciphertext, the decryption operation can be successful.

Recently, to achieve effective search as well as data access control over encrypted data, a few searchable encryption schemes with keyword search authorization are proposed [17]–[24]. However, the access structure in [17] only allows AND policies and these schemes in [18]–[20] use an LSSS (Linear Secret Sharing Scheme) matrix as an access policy and only consider AND and OR gates; schemes [21], [22] are built from the pairings in composite-order group with an impractical search cost; the scheme [23] also designed a novel attribute based keyword search scheme with dynamic access policy updating, but only supports AND and OR gates; Zhu et al. [24] proposed a key-policy attribute based encryption with equality test scheme, which supports AND, OR, and threshold gates. In contrary to [24], in this paper, we design a ciphertext-policy attribute-based searchable encryption with searchable capability that fulfills the full-fledged CP-ABE scheme proposed in [16], where a group with prime order is used and the access policy is denoted by the tree structure. Thus, our scheme supports ADN, OR, and threshold gates to flexibly express the access policy and has an acceptable search cost. The most similar work to ours has been proposed by Zheng *et al.* [1], however, our work outperforms their scheme on many aspects through experimental comparisons.

B. CONTRIBUTIONS

In this paper, our main contributions can be summarized as follows.

(1) We design a ciphtertext-policy attribute-based searchable encryption, which can achieve search and fine-grained access control over encrypted data simultaneously. Actually, through equipping the popular CP-ABE scheme [16] with the search capability, our design inherits all advantages of this scheme including security, flexible access policy expression, and fine-grained data access control.

(2) We provide detailed correctness analyses, performance analyses, and security proofs for our proposed CP-ABSE scheme.

(3) We implement our CP-ABSE scheme and the similar work CP-ABKS scheme proposed in [1]. Extensive experiments in a real data set demonstrate that our scheme outperforms CP-ABKS on many aspects.

II. RELATED WORK

A. SEARCHABLE ENCRYPTION

The first practical SSE was introduced by Song et al. [4], whose search time is linear in the size of the database. Goh [25] used Bloom Filter to construct secure index to improve the search time which is linear to the number of the documents. Another SSE scheme with linear search time is in [26] that can achieve forward privacy. Curtmola *et al.* [27] were first to formally define the security and leakage of SSE and proposed a new construction, achieving sub-linear search complexity by using the inverted index construction. Kamara *et al.* [5] deployed the dynamic feature for SSE with sub-linear search complexity. That is, the secure file deletion and addition are supported in their scheme. However, their scheme reveals hashes of the unique keywords contained in the document of the update. Kamara and Papamanthou's scheme overcomes the limitation by increasing the space of the used data structure. However, Stefanov *et al.* [28] claimed that these two schemes cannot achieve forward privacy, both of them leak information in the file updating processes, and proposed a practical dynamic SSE with small leakage. Recently, more efficient and secure dynamic SSE schemes were further studied in [7] and [8]. Considering that the server is the active adversary, verifiable SSE was proposed by Kurosawa and Ohtaki [6], which allows the search user to conduct correctness and completeness verification for search results. A popular application scenario of searchable encryption is secure search in the cloud computing environment, several multiple keyword ranked SSE schemes over encrypted cloud data were developed in [29]–[34].

Furthermore, schemes [33], [34] support advanced personalized searches with the high accuracy of the search results by using the user models. The first public-key based searchable encryption scheme was proposed by Boneh *et al.* [9]. Later, the conjunctive search, subset search, and range search were also explored in the public-key cryptosystem in [10]–[13]. Recently, researchers begin to use the emerging blockchain technique to design searchable encryption schemes such as [35] and [36]. These schemes can perfectly realize trusted and transparent verifiability of search results to resist malicious servers [37].

B. ATTRIBUTE-BASED ENCRYPTION

The concept of attribute-based encryption was introduced by Sahai and Water in [14]. In that scheme, a set of descriptive attributes are used to label a user's keys and a ciphertext. If and only if attributes in a key *match* attributes in a specified ciphertext, the key can decrypt the ciphertext. Due to realizing scalable and fine-grained access control over encrypted data, attribute-based encryption was further developed in [15], [16], and [38]–[43]. In these schemes, a set of attributes and an access structure are two key components. Constructions [15], [41]–[43] are referred to as KP-ABE, as the ciphertext is encrypted under a set of attributes and the user's private key is associated with a specified access policy. The decryption condition is that the attributes in the ciphertext have to satisfy the access policy in the user's private key. The CP-ABE schemes [16], [38]–[40] have exactly contrary configuration about attributes and an access policy, where a message is encrypted with an access policy and the user's key is associated with her attributes. If and only if the user's attributes satisfy the access policy in the ciphertext, the decryption can be conducted. In above all schemes, users' attributes and access policies are in plaintext form and thus reveal users' privacy information such as *sex, position, specialty* and so on. To address this problem, anonymous attribute-based encryption schemes were designed in [44]–[47]. These schemes can well preserve receivers' attribute privacy by hiding attribute information in ciphertexts [47]. We emphasize that, in this paper we only focus on the data and query privacy and the anonymous attribute-based searchable encryption will be our future work.

C. ATTRIBUTE-BASED SEARCHABLE ENCRYPTION

The first keyword authorized private search on public key encrypted data was proposed based on identity-based encryption by Camenisch *et al.* [48]. In that scheme, the data user needs to interact with the data owner to obtain search trapdoor. To enhance the expressiveness of access structure and flexibility of keyword search authorization, a few attribute-based searchable encryption schemes have been published [17]–[24]. In such a system, the data owner is able to flexibly control the data user's keyword search permissions in a fine-grained manner by using the promising attribute-based encryption primitive. However, the access structure in [17] only allows AND policies and these schemes in [18]–[20] use an LSSS (Linear Secret Sharing Scheme) matrix as an access policy that can only express AND and OR gates. Moreover, constructions in [21] and [22] are built from the pairings in composite-order group, which are more of theoretic interests because of impractical computational costs in a real search system. The scheme [23] also designed a novel attribute based keyword search scheme with dynamic access policy updating, but only supports AND and OR gates; Zhu *et al.* [24] proposed a key-policy attribute based encryption with equality test scheme, which supports AND, OR, and threshold gates. The most similar work to ours is proposed by Zheng *et al.* in [1], which is also constructed from the pairings in a prime order group and can flexibly support AND, OR, and threshold gates. However, our work outperforms their scheme on many aspects by our experimental comparisons.

III. PRELIMINARIES

In this section, we introduce several important definitions and techniques used to design our scheme.

A. BILINEAR PAIRING MAP

$\mathbb{G}_1$ and $\mathbb{G}_2$ denote two cyclic multiplicative groups with the identical order q. We define a map over $\mathbb{G}_1$ and $\mathbb{G}_2$ as $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. If map e satisfies the following properties, then e is a bilinear pairing map.

(1) Bilinear.

- For any $a, b \in \mathbb{Z}_q^*$ and $x, y \in \mathbb{G}_1$, $e(x^a, y^b) = e(x, y)^{ab}$ holds.
- For any $x_1, x_2, y \in \mathbb{G}_1$, $e(x_1x_2, y) = e(x_1, y)e(x_2, y)$ holds.
- For any $x, y_1, y_2 \in \mathbb{G}_1$, $e(x, y_1y_2) = e(x, y_1)e(x, y_2)$ holds.

(2) Computable. For any $x, y \in \mathbb{G}_1$, there exists a polynomial time algorithm to efficiently calculate $e(x, y) \in \mathbb{G}_2$.

(3) Non-degenerate. If g is a generator of $\mathbb{G}_1$, then $e(g, g)$ is a generator of $\mathbb{G}_2$.

B. ACCESS STRUCTURE AND ACCESS TREE

Definition 1: Access Structure. Let $P = \{P_1, P_2, \ldots, P_n\}$ *denote a set of parties. A collection* $\mathbb{A} \subseteq 2^P$ *is monotone if* $\forall B, C$*: if* $B \in \mathbb{A}$ *and* $B \subseteq C$ *then* $C \in \mathbb{A}$*. An access structure is a collection* $\mathbb{A}$ *of non-empty subsets of* $P = \{P_1, P_2, \ldots, P_n\}$*, i.e.,* $\mathbb{A} \subseteq 2^P \setminus \{\emptyset\}$*. The sets in* $\mathbb{A}$ *are called the authorized sets, and the sets not in* $\mathbb{A}$ *are called the unauthorized sets.*

Definition 2: Access Tree. Let T *be a tree representing an access control policy. In* T*, each non-leaf node represents a threshold gate, described by its children and a threshold value, and each leaf node represents an attribute. We use* num_x *and* k_x *to denote the number of children of a node* x *and its threshold value, respectively. There are three cases for the value of* k_x *if* x *is a non-leaf node: 1)* $k_x = 1$ *denotes node* x *is an OR gate; 2)* $k_x = num_x$ *means node* x *is an AND gate; 3)* $1 < k_x < num_x$ *denotes node* x *is a threshold gate. We define* $k_x = 1$ *when* x *is a leaf node.*

Several notations about access tree are defined as follows. *parent(x)* denotes the parent of x. For a leaf node x, we use *attr(x)* to denote the attribute associated with the leaf node. *index(x)* denotes the label of x. Given a node y with c children, its child nodes are numbered from 1 to c.

Satisfying an Access Tree: Let T_x be the subtree of T rooted at the node x. If a set of attributes S satisfies T_x, we denote it as $T_x(S) = 1$. $T_x(S)$ can be computed as follows. If x is a non-leaf node, evaluate $T_{x'}(S)$ for all children x' of node x. $T_x(S)$ return 1 if and only if at least k_x children return 1. If x is a leaf node, $T_x(S)$ returns 1 if and only if $attr(x) \in S$. Thus, according to the above recursive computation, if set S satisfies T, then $T_r(S) = 1$, where r is the root node of T.

C. SECURITY ASSUMPTION

Definition 3: Discrete Logarithm problem assumption (DL). Let $\mathbb{G}$ be a group with prime order q. g is a generator of $\mathbb{G}$. Given g and g^a, the DL problem assumption is defined as: no probabilistic polynomial-time adversary $\mathcal{A}$ can compute $a \in \mathbb{Z}_q^*$ with a non-negligible advantage.

Definition 4: Decisional Bilinear Diffie-Hellman(DBDH) problem assumption. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative groups with group q and e denote a bilinear pairing map. Given four elements $a, b, c, z \in \mathbb{Z}_q^*$ and $g^a, g^b, g^c \in \mathbb{G}_1, e(g, g)^{abc}, Z = e(g, g)^z \in \mathbb{G}_2$, DBDH problem assumption is defined as: no probabilistic polynomial-time adversary $\mathcal{A}$ can decide whether $Z = e(g, g)^{abc}$ or $e(g, g)^z$ with a non-negligible advantage.

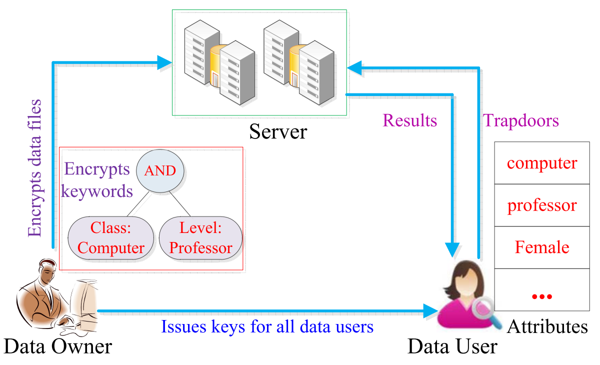


FIGURE 1. System model.

IV. SYSTEM MODEL AND SECURITY MODEL

A. SYSTEM MODEL

There are three entities in the system model, i.e., the data owner, multiple data users, and the server, as shown in Figure 1. To achieve the searchability over encrypted data files, the data owner uploads a secure searchable index, in which each index keyword is encrypted using a specified access control tree. When a data user applies to join in the system, the data owner generates a group of keys for the data user according to the user's attributes. The data user can use the issued keys to encrypt a search query to generate the legal search trapdoor. The server is responsible for performing search over the secure searchable index without knowing any information about data files and the search query. In the ciphertext-policy attribute-based searchable encryption, given an index keyword and a query, if and only if the attributes contained in the trapdoor satisfy the access tree encrypting the index keyword and the index keyword matches the query, a successful search can complete.

B. SECURITY MODEL

Generally, the security goal of an index based searchable encryption is that the adversary cannot obtain the keyword information from the searchable index and the query trapdoor. Thus, the security of data and the privacy of a query are protected in terms of the data owner and the search user, respectively. To formally evaluate the security, we give following two security definitions based on a game between a challenger $\mathcal{C}$ and a probabilistic polynomial-time adversary $\mathcal{A}$.

Definition 5: The index keyword unrecoverable security against chosen plaintext attack model (ikus-cpa).

$\mathcal{C}$ first gives system public parameters to $\mathcal{A}$ and $\mathcal{A}$ defines a challenge access tree T^*. $\mathcal{A}$ adaptively asks the challenger $\mathcal{C}$ for a group of private keys corresponding to attribute sets $S_1, S_2, \ldots, S_n$ and a group of ciphertexts corresponding to index keywords $k_1, \ldots, k_m$. There is a restriction that none of these responding private keys satisfy T^*. Then, $\mathcal{A}$ submits two keywords w_0 and w_1 to $\mathcal{C}$. $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$ and encrypts w_b to get the ciphertext $[w_b]$, which is sent to $\mathcal{A}$. $\mathcal{A}$ is allowed to continue to ask for private keys many times corresponding to attribute sets $S_{q+1}, S_{q+2}, \ldots$ and none of them satisfy T^*. Finally, $\mathcal{A}$ outputs b's guess b'. We define the advantage that $\mathcal{A}$ wins the game to be $Adv_{\mathcal{A}}^{ikus-cpa} = |Pr(b = b') - \frac{1}{2}|$. If $Adv_{\mathcal{A}}^{ikus-cpa}$ is negligible, we say that our proposed index keyword encryption achieves index keyword unrecoverable security against chosen plaintext attack model.

Definition 6: The query trapdoor unrecoverable security against eavesdropper attack model (qtus-eav).

$\mathcal{A}$ submits challenge keywords for many times to the the challenger $\mathcal{C}$ and each time $\mathcal{C}$ sends the corresponding ciphertext to $\mathcal{A}$ in response. Then, $\mathcal{A}$ sends two keywords w_0 and w_1 to $\mathcal{C}$, which are not challenged before. $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$ and encrypts w_b to get the ciphertext $[w_b]$, which is sent to $\mathcal{A}$. $\mathcal{A}$ is allowed to continue to ask $\mathcal{C}$ for the ciphertext of any keyword w, the only restriction is that w is not w_0 or w_1. Finally, $\mathcal{A}$ outputs b's guess b'. We define the advantage that $\mathcal{A}$ wins the game to be $Adv_{\mathcal{A}}^{qtus-eav} = |Pr(b = b') - \frac{1}{2}|$. If $Adv_{\mathcal{A}}^{qtus-eav}$ is negligible, we say that our proposed query keyword encryption achieves query trapdoor unrecoverable security against eavesdropper attack model.

We can see that *ikus-cpa* is allowed to access encryption oracle while *qtus-eav* is prohibited. This is because that our proposed index keyword encryption is the "probabilistic encryption", but the query keyword encryption is "deterministic encryption". Therefore, we use a weaker security model *qtus-eav* to define the security of query trapdoor, which is equivalent to the *Query Keyword Secrecy* model defined in [1] and [49].

In addition, we use the "curious but honest" threat model [50], where the server is the sole passive adversary. Assume that there exit secure communication channels, by which the data owner distributes keys to the authorized data user.

V. DEFINITION AND CONSTRUCTION

In this section, we first formally define the ciphertext-policy attribute-base searchable encryption scheme (CP-ABSE) and then describe its construction in detail.

A. DEFINITION

In this subsection, we define our proposed CP-ABSE at the system level as follows.

Definition 7: Our proposed CP-ABSE consists of the following five polynomial-time algorithms.

- Setup(1^λ) $\rightarrow$ (SSP, dk_1, dk_2). The algorithm is invoked by the data owner. It takes a security parameter λ as input and outputs system public parameters SPP and two keys dk_1 and dk_2. The data owner uses dk_1 to encrypt index keyword with an access tree and uses dk_2 to generate secret keys for a data user according to the data user's attributes.
- Keygen(SPP, dk_2, S_u) $\rightarrow \mathcal{K}_u$. The algorithm is invoked by the data owner and generates keys for a data user u with a set of attributes S_u. It takes as input system public parameter SSP, the key dk_2 and a data user's attribute set S_u, and outputs u's private key $\mathcal{K}_u$.
- Encind(SPP, dk_1, w, T_w) $\rightarrow \mathcal{I}_w$. The algorithm is run by the data owner and encrypts an index keyword. It takes as input system public parameter SSP, the key dk_1, an index keyword w, and an access tree T_w, and outputs w's ciphertext $\mathcal{I}_w$.
- Trpdr(SPP, $\mathcal{K}_u, q$) $\rightarrow \mathcal{T}_u$. The algorithm is run by the data user u. It takes as input the system public parameter SSP, a query keyword q, and u's private key $\mathcal{K}_u$, and outputs a trapdoor $\mathcal{T}_u(q)$, i.e., the search token with respect to q.
- Search(SPP, $\mathcal{I}_w, \mathcal{T}_u(q)$) $\rightarrow$ 1. The algorithm is run by the server. It takes as input the system public parameter SSP, an encrypted index keyword $\mathcal{I}_w$, and a trapdoor $\mathcal{T}_u(q)$ submitted by u, and outputs 1 if $w = q$ and u's attribute set S_u satisfies the access tree embedded in $\mathcal{I}_w$, simultaneously; otherwise, outputs 0.

Correctness: CP-ABSE scheme is correct if

$$\Pr\left[\begin{array}{l|l} \text{Search(SPP}, \mathcal{I}_w, & \text{Setup}(1^\lambda)\rightarrow (\text{SSP}, dk_1, dk_2); \\ \mathcal{T}_u(q)) \rightarrow 1 & \text{Keygen(SPP}, dk_2, S_u)\rightarrow \mathcal{K}_u; \\ & \text{Encind(SPP}, dk_1, w, T)\rightarrow \mathcal{I}_w; \\ & \text{Trpdr(SPP}, \mathcal{K}_u, q)\rightarrow \mathcal{T}_u. \end{array}\right] = 1$$

Algorithm 1 Setup

```
Input:
    Security parameter $\lambda$.
Output:
    System public parameter SPP, two keys $dk_1, dk_2$.
1: Generate two multiplicative groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime
   order $q$ and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. $g$ is a
   generator of $\mathbb{G}_1$
2: Generate two secure hash functions $H : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$
   and $H' : \{0,1\}^* \rightarrow \mathbb{G}_1$.
3: Define Lagrange coefficients
       $\triangle_{i,S}(x) = \prod_{j\in S, j\neq i} \frac{x-j}{i-j}$,
   where $S$ denotes a set and $i, j \in \mathbb{Z}_q^*$.
4: Randomly choose $\alpha, \beta \leftarrow \mathbb{Z}_q^*$.
5: Compute $g^\alpha$, $g^\beta$, and $e(g,g)^\alpha$.
6: SPP $\leftarrow (\mathbb{G}_1, \mathbb{G}_2, e, g, q, H, H')$.
7: $dk_1 \leftarrow (e(g,g)^\alpha, g^\beta)$, $dk_2 \leftarrow (\beta, g^\alpha)$
8: return SPP, $dk_1, dk_2$.
```

B. CONSTRUCTION

In this subsection, we present the implementation of each algorithm of CP-ABSE.

1) SETUP ALGORITHM

CP-ABSE Setup algorithm, as shown in Algorithm 1, is run by the data owner and creates running environment for CP-ABSE. By the algorithm, system public parameters and two keys are generated.

The public parameter can be used by all entities and dk_1 and dk_2 are used to encrypt index keywords to construct secure index and generate a data user's keys by the data owner, respectively.

Algorithm 2 Keygen

```
Input:
    Public parameter PPK, the key $dk_2$, and data user $u$'s
    attribute set $S_u$.
Output:
    $u$'s keys $\mathcal{K}_u$.
1: Randomly choose $r \leftarrow \mathbb{Z}_q^*$.
2: Compute $k_1 = g^{\frac{\alpha+r}{\beta}}$, $k_2 = g^{\frac{1}{\beta}}$, and $k_3 = g^r$.
3: for each attribute $a \in S_u$ do
4:     Randomly choose $r_a \leftarrow \mathbb{Z}_q^*$.
5:     Compute $k_a = k_3 \cdot H'(a)^{r_a}$ and $k'_a = g^{r_a}$.
6: end for.
7: return
       $\mathcal{K}_u = (k_1 = g^{\frac{\alpha+r}{\beta}}, k_2 = g^{\frac{1}{\beta}},$
           $\forall a \in S_u : k_a = g^r \cdot H'(a)^{r_a}, k'_a = g^{r_a})$
```

2) KEYGEN ALGORITHM

CP-ABSE Keygen algorithm, as shown in Algorithm 2, is run by the data owner and generates a group of keys for a data user u with attribute set S_u.

By this algorithm, the data owner generates $\mathcal{K}_u$ for data user u according to u's attributes, which is sent to u via secure communication channels. The data user u uses $\mathcal{K}_u$ to generate a legal query trapdoor for some interested query keyword.

3) ENCIND ALGORITHM

CP-ABSE Encind algorithm, as shown in Algorithm 3, is invoked by the data owner to encrypt index keywords to generate secure searchable index, where each index keyword is embedded an access tree.

Algorithm 3 Encind

```
Input:
    Public parameter PPK, the key $dk_1$, an index keyword $w$, and an access tree $T_w$.
Output:
    $w$'s encryption version $\mathcal{I}_w$.
1:  Randomly choose a secret value $s \leftarrow \mathbb{Z}_q^*$.
2:  Compute $\mathcal{I}'_w = e(g^{H(w)s}, g)e(g, g)^{\alpha s}$ and $\mathcal{I}''_w = g^{\beta s}$.
3:  Let $t$ be the root node of $T_w$.
4:  for each node $x$ in $T_w$ (in a top-down manner, starting from $t$) do
5:      if $x$ is the root node $t$ then
6:          Randomly choose a polynomial $q_t$ for $t$ with degree $d_t = k_t - 1$ and sets $q_t(0) = s$.
7:          Randomly set $d_t$ other points of $q_t$ to completely define it.
8:      else
9:          Randomly choose a polynomial $q_x$ for $x$ with degree $d_x = k_x - 1$.
10:         Set $q_x(0) = q_{parent(x)}(index(x))$.
11:         Randomly choose $d_x$ other points to completely define $q_x$.
12:     end if
13: end for
14: Let $X$ be the set of leaf nodes.
15: for each leaf node $x$ in $X$ do
16:     Compute $X_x = g^{q_x(0)}$ and $X'_x = H'(attr(x))^{q_x(0)}$
17: end for
18: return
        $\mathcal{I}_w = (T_w, \mathcal{I}'_w = e(g^{H(w)s}, g)e(g, g)^{\alpha s}, \mathcal{I}''_w = g^{\beta s}, \forall x \in X : X_x = g^{q_x(0)}, X'_x = H'(attr(x))^{q_x(0)})$
```

The data owner invokes this algorithm to encrypt all index keywords, each index keyword corresponds to an access tree that defines the search permission of the index keyword. For achieving sub-linear search complexity, we organize encrypted index keywords and encrypted data files as inverted index construction by using an array and a look-up table data structures, please refer to [27].

4) TRPDR ALGORITHM

CP-ABSE Trpdr algorithm, as shown in Algorithm 4, is used to encrypt a query keyword by a data user u with key $\mathcal{K}_u$.

Algorithm 4 TrpDr

```
Input:
    $u$'s key $\mathcal{K}_u$ and a search query $q$.
Output:
    $q$'s trapdoor $\mathcal{T}_u(q)$.
1: Compute $k_2^{H(q)} = g^{\frac{H(q)}{\beta}}$ and $k_1 \cdot g^{\frac{H(q)}{\beta}} = g^{\frac{\alpha+r+H(q)}{\beta}}$
2: return
       $\mathcal{T}_u(q) = (T = g^{\frac{\alpha+r+H(q)}{\beta}}, \forall a \in S_u : t_a = k_a = g^r \cdot H'(a)^{r_a}, t'_a = k'_a = g^{r_a})$
```

After the data user u encrypts a query keyword q, the query trapdoor $\mathcal{T}_u(q)$ is sent to the server.

5) SEARCH ALGORITHM

The server invokes CP-ABSE Search algorithm, as shown in Algorithm 5, to perform search over encrypted inverted index according to a query trapdoor submitted by a data user. In the whole processes of search, no useful information about data files and the query is revealed to the server.

According to Algorithm 5, we can observe that, given an encrypted index keyword $\mathcal{I}_w$ and a query trapdoor $\mathcal{T}_u(q)$ submitted by u, if and only if the two conditions: (1) "*the query keyword q is equal to the index keyword w, q=w*" and (2) "*u's attribute set S_u satisfies the access tree T_w*" hold simultaneously, the algorithm return 1. On the other hand, there exist two cases that Algorithm 5 returns 0. One is that u's attribute set S_u does not satisfy the access tree in $\mathcal{I}_w$, the algorithm will terminate in advance, which means u has not search permissions with respect to the index keyword w. The other is that u has the search permissions of index keyword w, however, the query q is not identical to w.

VI. CP-ABSE ANALYSIS

In this section, we provide detailed analyses for our proposed CP-ABSE, including correctness analysis, algorithm complexity analysis, and security analysis.

A. CORRECTNESS ANALYSIS

We first analyze the correctness of matching between an encrypted index keyword and a trapdoor.

Algorithm 5 tells us that, given $\mathcal{I}_w$ and $\mathcal{T}_u(q)$, if u's attributes satisfy the access tree T_w in $\mathcal{I}_w$, the sever can judge whether q is equal to w or not by checking whether Equation (1) is true or not:

$$\mathcal{I}'_w = \frac{e(\mathcal{I}''_w, T)}{F_t} \tag{1}$$

We can verify the correctness by following deduction:

$$\frac{e(\mathcal{I}''_w, T)}{F_t} = \frac{e(g^{\beta s}, g^{\frac{\alpha+r+H(q)}{\beta}})}{e(g, g)^{rs}} = \frac{e(g^s, g^{\alpha+r+H(q)})}{e(g, g)^{rs}}$$

Algorithm 5 Search

```
Input:
    An encrypted index keyword $\mathcal{I}_w$ and a query $q$'s trapdoor
    $\mathcal{T}_u$ submitted by $u$ with attribute set $S_u$.
Output:
    1, if $q = w$ and $S_u$ satisfies $T_w$; 0, otherwise.
1:  Let $x$ be a node of $T_w$.
2:  for each leaf node $x$ in $T_w$ do
3:      Let $a$ denote the attribute associated with the leaf node
        $x$, i.e., $a = attr(x)$.
4:      if $a \in S_u$ then
5:          Compute $F_x = \frac{e(t_a, X_x)}{e(t'_a, X'_x)} = \frac{e(g^r \cdot H'(a)^{r_a}, g^{q_x(0)})}{e(g^{r_a}, H'(a)^{q_x(0)})} = e(g,g)^{rq_x(0)}$
6:      else
7:          Define $F_x = \perp$.
8:      end if
9:  end for
10: for each non-leaf node $x$ in $T_w$ (in a down-top manner)
    do
11:     Let $S_x$ denote an arbitrary $k_x$-sized set of child nodes $z$
        such that $F_z \neq \perp$
12:     if no such set exists then
13:         Define $F_x = \perp$.
14:     else
15:         Compute (using Lagrange interpolation)
            $F_x = \prod_{z \in S_x} F_z^{\Delta_{i,S'_x}(0)}$
            $= \prod_{z \in S_x} (e(g,g)^{r \cdot q_z(0)})^{\Delta_{i,S'_x}(0)}$
            $= \prod_{z \in S_x} (e(g,g)^{r \cdot q_{parent(z)}(index(z))})^{\Delta_{i,S'_x}(0)}$
            $= \prod_{z \in S_x} e(g,g)^{r \cdot q_x(i) \cdot \Delta_{i,S'_x}(0)}$
            $= e(g,g)^{r \cdot q_x(0)}$
            where $i = index(z)$, $S'_x = (\forall z \in S_x : index(z))$, and
            $\Delta_{i,S'_x}$ is the Lagrange coefficient.
16:     end if
17: end for
18: Let $t$ be the root node of $T_w$.
19: if $F_t = \perp$ then
20:     return 0.
21:     (This means access tree $T_w$ is not satisfied by $u$'s
        attributes $S_u$.)
22: else
23:     Recursively compute $F_t = e(g,g)^{rq_t(0)} = e(g,g)^{rs}$.
24:     Compute $C = \frac{e(\mathcal{I}''_w, T)}{F_t}$
25:     if $C$ is equal to $\mathcal{I}'_w$ then
26:         return 1.
27:     else
28:         return 0.
29:     end if
30: end if
```

TABLE 1. Notations used in complexity analysis.

Notations	Description
P	Time of a pairing computation
$M_{\mathbb{G}_1}$	Time of a multiplication in group $\mathbb{G}_1$
$M_{\mathbb{G}_2}$	Time of a multiplication in group $\mathbb{G}_2$
$E_{\mathbb{G}_1}$	Time of an exponentiation in group $\mathbb{G}_1$
$E_{\mathbb{G}_2}$	Time of an exponentiation in group $\mathbb{G}_2$
$V_{\mathbb{G}_2}$	Time of choosing the inverse of an element in $\mathbb{G}_2$
$M_{\mathbb{Z}_q^*}$	Time of a multiplication in $\mathbb{Z}_q^*$
$A_{\mathbb{Z}_q^*}$	Time of a addition in $\mathbb{Z}_q^*$
$S_{\mathbb{Z}_q^*}$	Time of a subtraction in $\mathbb{Z}_q^*$
$V_{\mathbb{Z}_q^*}$	Time of choosing the inverse of an element in $\mathbb{Z}_q^*$
$\mathbb{Z}_q^*$	Time of choosing an element from $\mathbb{Z}_q^*$
H	Time of a hash computation, $H : \{0,1\}^* \to \mathbb{Z}_q^*$
H'	Time of a hash computation, $H' : \{0,1\}^* \to \mathbb{G}_1$
$\|x\|$	Cardinality of a set x.
$\|\|y\|\|$	Bit length of an element y.
X	Leaf-node set of an access tree
S	Attribute set of a data user
N	Least attribute set satisfying an access tree

$$
\begin{aligned}
&= \frac{e(g^s, g^\alpha)e(g^s, g^r)e(g^s, g^{H(q)})}{e(g,g)^{rs}} \\
&= e(g^s, g^{H(q)})e(g^s, g^\alpha) \\
&= e(g^{H(q)s}, g)e(g,g)^{\alpha s} \\
&\overset{w=q}{=\!=} \mathcal{I}'_w
\end{aligned}
\qquad (2)
$$

B. COMPLEXITY ANALYSIS

In this subsection, we theoretically analyze the time complexity of our proposed CP-ABSE scheme and Zheng *et al*'s CP-ABKS scheme in [1], which also provides an explicit and convincing performance comparison between the two schemes. Firstly, we define some necessary notations, as shown in Table 1.

For ease of reading, we describe the computation cost and the output size of each algorithm in CP-ABSE and CP-ABKS in Table 2 and Table 3, respectively. Note that we do not consider the communication cost of transmitting search results for a successful query, thus the output size of Search algorithm is set to be 0.

Since the pairing and the exponentiation computation are relatively consuming-time operation in a bilinear group, we can observe from Table 2 and Table 3 that CP-ABSE outperforms CP-ABKS on the key generation, the trapdoor generation, and the search complexity due to requiring the less pairing and exponentiation operations.

C. SECURITY ANALYSIS

In this subsection, we prove that the security of CP-ABSE in the *ikus-cpa* and *qtus-eav* model.

Theorem 1: If DBDH problem is intractable, our proposed index keyword encryption algorithm achieves index keyword unrecoverable security against chosen plaintext attack model (ikus-cpa).

TABLE 2. **The computation cost and output size of CP-ABSE.**

Algorithm	Computation Cost	Output Size
Setup	$P + 2E_{\mathbb{G}_1} + 2\mathbb{Z}_q^*$	$\|\mathbb{Z}_q^*\| + 3\|\|\mathbb{G}_1\|\| + \|\|\mathbb{G}_2\|\|$
Keygen	$(\|S\|+1)\mathbb{Z}_q^* + \|S\|H' + (2\|S\|+4)E_{\mathbb{G}_1} + V_{\mathbb{Z}_q^*} + M_{\mathbb{Z}_q^*} + (\|S\|+1)M_{\mathbb{G}_1}$	$(2\|S\|+2)\|\|\mathbb{G}_1\|\|$
Encind	$\mathbb{Z}_q^* + M_{\mathbb{Z}_q^*} + (2\|X\|+2)E_{\mathbb{G}_1} + P + H + E_{\mathbb{G}_2} + M_{\mathbb{G}_2} + \|X\|H'$	$(2\|X\|+1)\|\|\mathbb{G}_1\|\| + \|\|\mathbb{G}_2\|\|$
Trpdr	$E_{\mathbb{G}_1} + M_{\mathbb{G}_1} + H$	$(2\|S\|+1)\|\|\mathbb{G}_1\|\|$
Search	$(2\|N\|+1)P + (\|N\|+1)V_{\mathbb{G}_2} + (\|N\|+1)M_{\mathbb{G}_2} + \|N\|E_{\mathbb{G}_2}$	0

TABLE 3. **The computation cost and output size of CP-ABKS in [1].**

Algorithm	Computation Cost	Output Size
Setup	$3E_{\mathbb{G}_1} + 3\mathbb{Z}_q^*$	$3\|\mathbb{Z}_q^*\| + 4\|\|\mathbb{G}_1\|\|$
Keygen	$(\|S\|+1)\mathbb{Z}_q^* + \|S\|H' + (3\|S\|+1)E_{\mathbb{G}_1} + S_{\mathbb{Z}_q^*} + V_{\mathbb{Z}_q^*} + 2M_{\mathbb{Z}_q^*} + \|S\|M_{\mathbb{G}_1}$	$(2\|S\|+1)\|\|\mathbb{G}_1\|\|$
Encind	$2\mathbb{Z}_q^* + M_{\mathbb{Z}_q^*} + A_{\mathbb{Z}_q^*} + (2\|X\|+4)E_{\mathbb{G}_1} + H + M_{\mathbb{G}_1} + \|X\|H'$	$(2\|X\|+3)\|\|\mathbb{G}_1\|\|$
Trpdr	$H + M_{\mathbb{G}_1} + (2\|S\|+4)E_{\mathbb{G}_1}$	$(2\|S\|+3)\|\|\mathbb{G}_1\|\|$
Search	$(2\|N\|+3)P + \|N\|V_{\mathbb{G}_2} + (\|N\|+2)M_{\mathbb{G}_2} + \|N\|E_{\mathbb{G}_2}$	0

Proof: Assume that there exists a probabilistic polynomial time adversary $\mathcal{A}$ can recovery the index keyword information from the index keyword encryption with a non-negligible advantage ϵ, we can construct a simulator $\mathcal{B}$ to solve the DBDH problem with the non-negligible advantage $\frac{\epsilon}{2}$.

The challenger $\mathcal{C}$ randomly chooses a bit $\nu \in \{0, 1\}$ and sends t_ν to the simulator $\mathcal{B}$, where $t_0 = (g, A = g^a, B = g^b, C = g^c, Z = e(g, g)^{abc})$, $t_1 = (g, A = g^a, B = g^b, C = g^c, Z = e(g, g)^z)$, a, b, c, z are random elements in $\mathbb{Z}_q^*$. The simulator $\mathcal{B}$ plays the following game with the adversary $\mathcal{A}$ based on *ikus-cpa*.

(1) The simulator $\mathcal{B}$ computes a public parameter $Y = e(B, C) = e(g, g)^{bc}$ and sends it to the adversary $\mathcal{A}$. $\mathcal{A}$ defines an access tree T^* that he wants to be challenged.

(2) $\mathcal{A}$ adaptively asks $\mathcal{B}$ for private keys $\mathcal{KA}^{S_1}, \mathcal{KA}^{S_2}, \ldots, \mathcal{KA}^{S_n}$ of attribute sets $S_1, S_2, \ldots, S_n$ and the ciphertexts $\mathcal{I}_{k_1}, \mathcal{I}_{k_2}, \ldots, \mathcal{I}_{k_m}$ of index keywords $k_1, k_2, \ldots, k_m$. These private keys and ciphertexts responded by $\mathcal{B}$ satisfy the following three conditions:

- All attribute sets $S_1, S_2, \ldots, S_n$ embedded into the corresponding private keys do not satisfy the challenge access tree T^*.
- All private keys can be used to generate legal query trapdoor.
- Given a private key $\mathcal{KA}^{S_i}$, $i \in [1, n]$ and a query keyword q, the generated query trapdoor is denoted as $\mathcal{KA}^{S_i}(q)$. There exists an index keyword ciphertext $\mathcal{I}_{k_j}, j \in [1, m]$ that the algorithm Search(SSP, $\mathcal{I}_{k_j}, \mathcal{KA}^{S_i}(q)$) $= 1$, i.e., $q = k_j$ and the attribute set S_i satisfies the access tree T_{k_j}.

(3) The adversary submits two index keywords w_0 and w_1 that he wishes to be challenged and the access tree T^* to $\mathcal{B}$. $\mathcal{B}$ randomly chooses a bit $b \in \{0, 1\}$ and generates the following ciphertext:

$$\mathcal{I}_{w_b}^* = (T^*, (\mathcal{I}'_{w_b})^* = e(A^{H(w_b)}, g)Z, (\mathcal{I}''_w)^* = A^\beta, \\ \forall x \in X^* : X_x^* = g^{q_x(0)}, (X'_x)^* = H'(attr(x))^{q_x(0)}),$$

where β is randomly chosen from $\mathbb{Z}_q^*$ and X^* denotes the leaf node set of T^*. Finally, $\mathcal{B}$ sends $\mathcal{I}_{w_b}^*$ to $\mathcal{A}$.

(4) $\mathcal{A}$ continues to adaptively ask for private keys $\mathcal{KA}^{S_{n+1}}, \mathcal{KA}^{S_{n+2}}, \ldots$, corresponding to attribute sets S_{n+1}, $S_{n+2}, \ldots$, respectively, and the ciphertexts $\mathcal{I}_{k_{m+1}}, \mathcal{I}_{k_{m+2}}, \ldots$ of index keywords $k_{m+1}, k_{m+2}, \ldots$, with the restriction that none of these attribute sets $S_{n+1}, S_{n+2}, \ldots$, satisfy T^*.

(5) $\mathcal{A}$ outputs the guess b' of b. Obviously, the adversary cannot correctly decide $b = 0$ or $b = 1$ by letting the search algorithm Search(SSP, $\mathcal{I}_{w_b}^*, \mathcal{KA}^{S_i}(w_0)$) or Search(SSP, $\mathcal{I}_{w_b}^*$, $\mathcal{KA}^{S_i}(w_1)$) output 1, since none of the attribute sets queried by $\mathcal{A}$ satisfy the access tree T^*. Therefore, the adversary $\mathcal{A}$ has to recover the index keyword information $H(w_b)$ from $\mathcal{I}_{w_b}^*$ to decide $b = 0$ or $b = 1$. There are two conditions as below:

- If $\nu = 0$, then t_0 is sent to $\mathcal{B}$ and $Z = e(g, g)^{abc}$ and

$$\mathcal{I}_{w_b}^* = (T^*, (\mathcal{I}'_{w_b})^* = e(g^{aH(w_b)}, g)e(g, g)^{abc}, \\ (\mathcal{I}''_w)^* = g^{a\beta}, \quad \forall x \in X^* : X_x^* = g^{q_x(0)}, \\ (X'_x)^* = H'(attr(x))^{q_x(0)}).$$

Since α and s are randomly chosen in the index keyword encryption, we let $a = s$ and $bc = \alpha$, the cipertext can be denoted as

$$\mathcal{I}_{w_b}^* = (T^*, (\mathcal{I}'_{w_b})^* = e(g^{sH(w_b)}, g)e(g, g)^{\alpha s}, \\ (\mathcal{I}''_w)^* = g^{s\beta}, \quad \forall x \in X^* : X_x^* = g^{q_x(0)}, \\ (X'_x)^* = H'(attr(x))^{q_x(0)}).$$

This means that $\mathcal{I}_{w_b}^*$ is the correct ciphertext of index keyword w_b.

- If $\nu = 1$, then t_1 is sent to $\mathcal{B}$ and $Z = e(g, g)^z$ and the cipertext can be denoted as

$$\mathcal{I}_{w_b}^* = (T^*, (\mathcal{I}'_{w_b})^* = e(g^{sH(w_b)}, g)e(g, g)^z, (\mathcal{I}''_w)^* = g^{s\beta}, \\ \forall x \in X^* : X_x^* = g^{q_x(0)}, (X'_x)^* = H'(attr(x))^{q_x(0)}).$$

Since z is a random element, $\mathcal{I}_{w_b}^*$ is also a random element from the view of the adversary $\mathcal{A}$ and contains no information about w_b.

$\mathcal{A}$ outputs b's guess b'. If $b = b'$, then $\mathcal{B}$ outputs ν's guess $\nu' = 0$, the challenger $\mathcal{C}$ sends the valid encryption parameter t_0 to $\mathcal{B}$. Since $\mathcal{A}$ has advantage ϵ to recover $H(w_b)$ from $\mathcal{I}^*_{w_b}$, the probability that $\mathcal{A}$ outputs $b' = b$ is $\frac{1}{2} + \epsilon$. If $b \neq b'$, then $\mathcal{B}$ outputs ν's guess $\nu' = 1$ and the challenger $\mathcal{C}$ sends the random encryption parameter t_1 to $\mathcal{B}$. The probability that $\mathcal{A}$ outputs $b' = b$ is $\frac{1}{2}$.

The overall advantage that $\mathcal{B}$ solves the DBDH problem in the above game is as follows:

$$\left|\frac{1}{2}Pr[\nu = \nu'|\nu = 0] + \frac{1}{2}Pr[\nu = \nu'|\nu = 1] - \frac{1}{2}\right| = \left|\left[\frac{1}{2}\left(\frac{1}{2} + \epsilon\right) + \frac{1}{2}\cdot\frac{1}{2}\right] - \frac{1}{2}\right| = \frac{\epsilon}{2} \tag{3}$$

Since ϵ is non-negligible, therefore $\frac{\epsilon}{2}$ is also non-negligible such that $\mathcal{B}$ can solve the DBDH problem with a non-negligible advantage, which contradicts DBDH problem assumption. □

Theorem 2: If DL problem is intractable, our proposed query keyword encryption achieves query trapdoor unrecoverable security against eavesdropper attack model (qtus-eav).

Proof: We prove the above theorem by the following game between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$.

(1) The adversary $\mathcal{A}$ submits queries $k_1, k_2, \ldots, k_n$ to the challenger $\mathcal{C}$ many times. For each k_i, $1 \leq i \leq n$, $\mathcal{C}$ sends the following ciphertext to $\mathcal{A}$ in response:

$$\mathcal{T}_{\mathcal{A}}(qi) = (T = g^{\frac{\alpha+r+H(qi)}{\beta}}, \forall a \in S_{\mathcal{A}} : t_a = g^r \cdot H'(a)^{r_a}, (t'_a) = g^{r_a}),$$

where we use $S_{\mathcal{A}}$ to denote the adversary $\mathcal{A}$'s attribute set.

(2) $\mathcal{A}$ sends two challenge query keywords q_0 and q_1 to $\mathcal{C}$ with the restriction that q_0 and q_1 have never been queried before.

(3) $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$ and encrypts q_b to be

$$\mathcal{T}^*_{\mathcal{A}}(q_b) = (T^* = g^{\frac{\alpha+r+H(q_b)}{\beta}}, \forall a \in S_{\mathcal{A}} : t^*_a = g^r \cdot H'(a)^{r_a}, (t'_a)^* = g^{r_a}),$$

and sends $\mathcal{T}^*_{\mathcal{A}}(q_b)$ to $\mathcal{A}$.

(4) $\mathcal{A}$ continues to ask the challenger $\mathcal{C}$ for the ciphertext of any query keyword other than q_0 and q_1.

(5) $\mathcal{A}$ outputs the guess b' of b. Since the adversary $\mathcal{A}$ is not allowed to access the encryption oracle, it cannot effectively compute ciphertexts $\mathcal{T}^*_{\mathcal{A}}(q_0)$ and $\mathcal{T}^*_{\mathcal{A}}(q_1)$ without α, r, and β. Thus, the probability that $\mathcal{A}$ outputs the guess $b' = b$ is at most $\frac{1}{2}$ as long as the DL problem is intractable.

If solving the DL problem is computationally feasible in polynomial time, the adversary can output $b' = b$ with probability 1 by finding out correct encryption parameter $\alpha + r$ and β. For example, $\mathcal{A}$ first arbitrarily chooses two response ciphertexts $g^{\frac{\alpha+r+H(qi)}{\beta}}$ and $g^{\frac{\alpha+r+H(qj)}{\beta}}$ of queries qi and qj (Hash values $H(qi), H(qj)$ can be efficiently calculated).

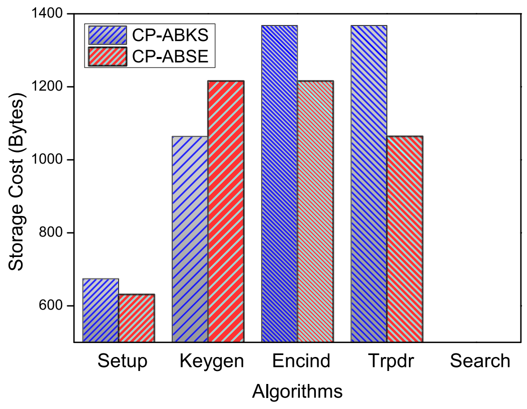


FIGURE 2. The storage cost of each algorithm.

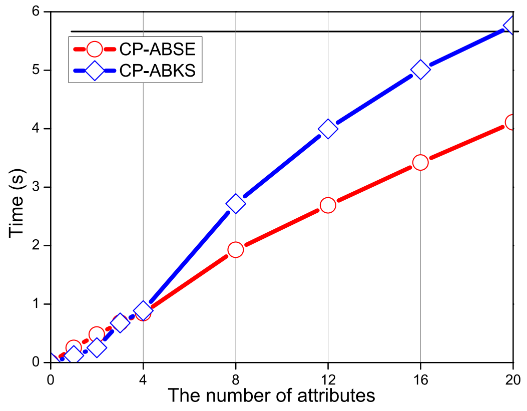


FIGURE 3. The time cost of keygen algorithm.

Then, $\mathcal{A}$ solves DL to set up the following equation set to obtain $\alpha + r$ and β:

$$\begin{cases} t'\beta = \alpha + r + H(qi) \\ t''\beta = \alpha + r + H(qj) \end{cases} \tag{4}$$

$$\begin{cases} \beta = \dfrac{H(qi) - H(qj)}{t' - t''} \\ \alpha + r = \dfrac{t''H(qi) - t'H(qj)}{t' - t''} \end{cases} \tag{5}$$

As a result, the adversary $\mathcal{A}$ can compute the valid ciphertext components $g^{\frac{\alpha+r+H(q0)}{\beta}}$ and $g^{\frac{\alpha+r+H(q1)}{\beta}}$ of the challenge query keywords q_0 and q_1. □

VII. EXPERIMENTAL EVALUATION

In this section, we test the practical performance of our proposed CP-ABSE through experimental evaluation on a real data set, Request For Comments Database (RFC) [51]. Also, we implement CP-ABKS scheme proposed in [1] for providing a performance comparison with our scheme. The experimental results demonstrate that our scheme has appreciable performance advantages on many aspects compared with CP-ABKS.

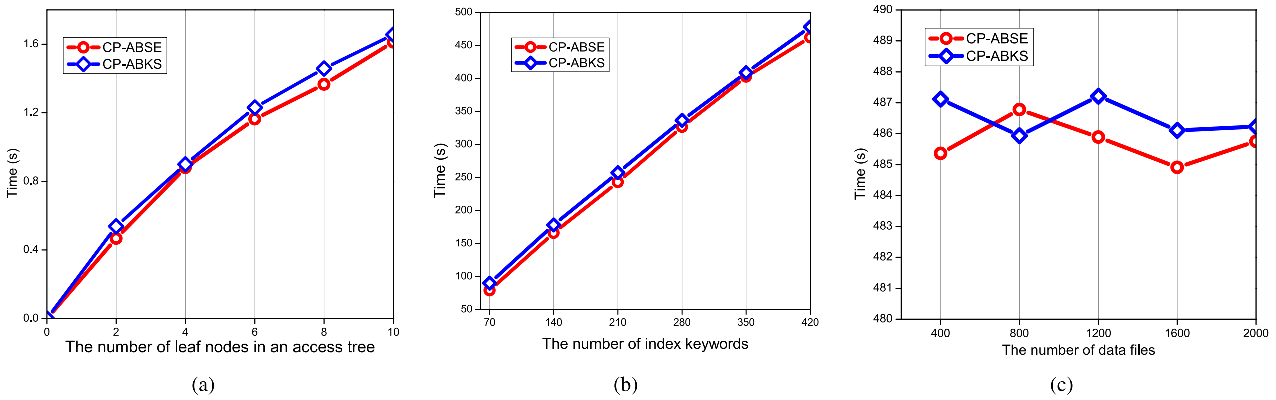


FIGURE 4. Time cost of index keyword encryption. (a) Time cost of encrypting one index keyword for different number of leaf nodes. (b) Time cost of encrypting index keywords for different number of index keywords with fixed number of leaf nodes $|X| = 6$ and fixed number of data files $n = 2000$. (c) Time cost of encrypting index keywords for different number of data files with fixed number of leaf nodes $|X| = 6$ and fixed number of index keyword $i = 420$.

A. SETUP

The experimental environments are classified into the server side and the client side. The server side is a Windows 7 desktop system with 3.60-GHz Intel Core (TM) i7-7700 CPU and 8-GB RAM, which plays the role of the server to perform search over encrypted index keywords. The client side is a Windows 7 desktop system with 2.3-GHz Intel Core (TM) i5-6200U and 4-GB RAM, which is used to test the time cost of index keyword generation and trapdoor generation.

We randomly choose 2000 data files from RFC as our experimental data set, where 420 index keywords are extracted by using Hermetic Word Frequency Counter [52]. All programs are developed in Java language on JPBC library [53], where symmetric elliptic curve group *Type A* with prime order is chosen in the whole experiments.

B. EVALUATION OF THE STORAGE COST

Fig. 2 shows the output size of each algorithm in CP-ABSE and CP-ABKS. In the experiment, we set $|S|$ and $|X|$ to be 3 uniformly. From Fig. 2, we can see that, compared with CP-ABKS, our CP-ABSE scheme needs lesser spaces to storage system parameters and keys for the data owner, the query trapdoor for the data user, and the encrypted index keywords for the server. In the practical query, the Search algorithm in two schemes only outputs 1 or 0 and the storage cost is almost zero.

C. EVALUATION OF KEYGEN ALGORITHM

Keygen algorithm generates private keys for an authorized data user. Fig. 3 shows the time cost of running Keygen algorithm. We can observe that the time cost of the two schemes on private key generation is linear to the number of the data user's attributes. The more number of attributes the data user has, the more time are expended on private key generation. We can also see that our proposed CP-ABSE needs lesser time when the number of attributes is greater than 2, since lesser exponentiation operations are required in CP-ABSE compared with CP-ABKS.

D. EVALUATION OF ENCIND ALGORITHM

Encind Algorithm outputs the encrypted version of an index keyword, which is embedded an access tree with $|X|$ leaf nodes. Fig. 4(a) shows that the time cost of one index keyword encryption of CP-ABSE and CP-ABKS, which is linear to the number of leaf nodes in the access tree. From Fig. 4(b) and 4(c), we can see that, for CP-ABSE and CP-ABKS, the time cost of the whole secure index construction is linearly related to the number of index keywords extracted from data files while has nothing to do with the size of data file collection when fixing the number of leaf nodes in access trees. We can also see that the two schemes have approximately equal performance cost on secure index construction. For example, encrypting 420 index keywords to construct secure index needs to expend about 486 seconds in our experiment.

E. EVALUATION OF TRPDR ALGORITHM

Trpdr algorithm is run by an authorized data user and outputs a trapdoor of a query keyword. Fig. 5 shows the time cost of encrypting a query keyword to generate query trapdoor for schemes CP-ABSE and CP-ABKS. We can observe that the time cost of trapdoor generation in CP-ABSE is not affected by the number of the data user's attributes, while linearly increases with an increasing number of attributes in CP-ABKS scheme.

F. EVALUATION OF SEARCH ALGORITHM

The server runs Search algorithm to perform search over encrypted index keywords according to a query trapdoor. In our experiment, we organize the encrypted index keywords and encrypted data files as inverted index construction. The linear search is used for keyword match between the inverted index and the query trapdoor. To enhance the testing stress, we generate a query trapdoor that satisfies all access trees encrypting index keywords. Fig. 6 shows the average time cost of Search algorithm, where N denotes the least attribute

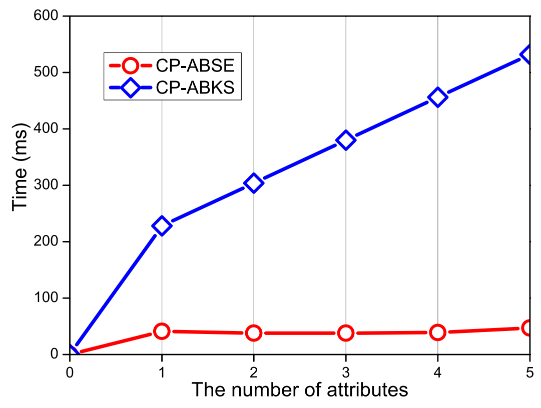
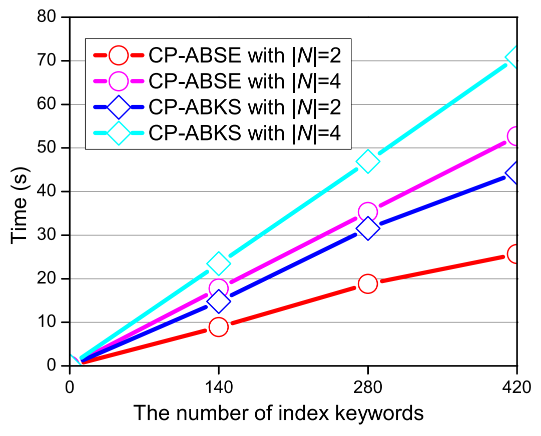
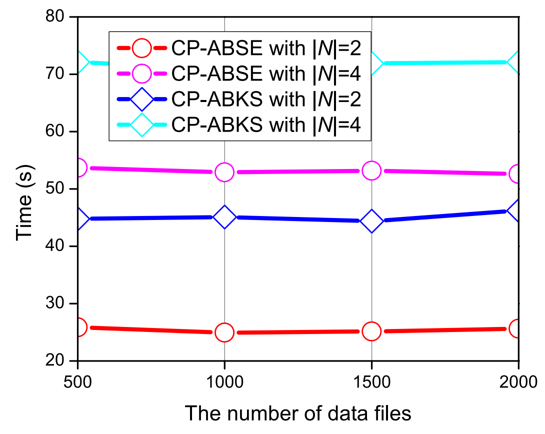


FIGURE 5. The time cost of trapdoor generation.



(a)



(b)

FIGURE 6. Time cost of search over encrypted data. (a) Time cost of search for different number of index keywords with fixed size of data files $n = 2000$. (b) Time cost of search for different size of data files with fixed number of index keyword $i = 420$.

set satisfying an access tree. The experiment results tell us that the time cost of Search algorithm in CP-ABSE and CP-ABKS is linearly related with the number of index keywords and the least attribute set satisfying an access tree, and is not affected by the number of data files. More importantly, we can see from Fig. 6(a) and Fig. 6(b) that, compared with CP-ABKS, our scheme has the better search performance, which is a key evaluation index in a search system.

VIII. CONCLUSION

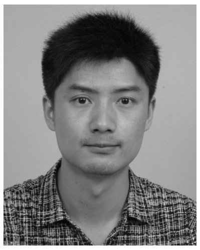
In this paper, we propose a ciphertext-policy attribute-based searchable encryption scheme, CP-ABSE. The scheme can achieve keyword based search and fine-grained access control over encrypted data, simultaneously. We provide detailed performance and security analyses for our scheme. Also, we implement our proposed CP-ABSE and a similar work CP-ABKS proposed in [1]. Experimental results demonstrate that our scheme has the better search performance compared with CP-ABKS. The dynamic, forward secure, and anonymous attributed-based searchable encryption scheme is our future work.

REFERENCES

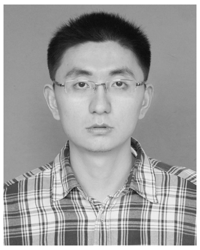
[1] Q. Zheng, S. Xu, and G. Ateniese, “VABKS: Verifiable attribute-based keyword search over outsourced encrypted data,” in *Proc. IEEE INFOCOM*, May 2014, pp. 522–530.
[2] K. Ren, C. Wang, and Q. Wang, “Security challenges for the public cloud,” *IEEE Internet Comput.*, vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.
[3] S. Kamara and K. Lauter, “Cryptographic cloud storage,” in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, 2010.
[4] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in *Proc. IEEE Symp. Secur. Privacy*, vol. 8, May 2000, pp. 44–55.
[5] S. Kamara, C. Papamanthou, and T. Roeder, “Dynamic searchable symmetric encryption,” in *Proc. ACM CCS*, 2012, pp. 965–976.
[6] K. Kurosawa and Y. Ohtaki, “UC-secure searchable symmetric encryption,” in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, 2012, pp. 285–298.
[7] K. S. Kim, M. Kim, D. Lee, J. H. Park, and W. H. Kim, “Forward secure dynamic searchable symmetric encryption with efficient updates,” in *Proc. ACM CCS*, 2017, pp. 1449–1463.
[8] R. Bost, B. Minaud, and O. Ohrimenko, “Forward and backward private searchable encryption from constrained cryptographic primitives,” in *Proc. ACM CCS*, 2017, pp. 1465–1482.
[9] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in *Proc. EUROCRYPR*, 2004, pp. 506–522.
[10] P. Golle, J. Staddon, and B. Waters, “Secure conjunctive keyword search over encrypted data,” in *Applied Cryptography and Network Security*. Berlin, Germany: Springer, 2004, pp. 31–45.
[11] L. Ballard, S. Kamara, and F. Monrose, “Achieving efficient conjunctive keyword searches over encrypted data,” in *Proc. IEEE ICICS*, Jun. 2005, pp. 414–426.
[12] D. Boneh and B. Waters, “Conjunctive, subset, and range queries on encrypted data,” in *Theory of Cryptography*. Springer, Germany: Springer, 2007, pp. 535–554.
[13] H. Yin *et al.*, “Secure conjunctive multi-keyword search for multiple data owners in cloud computing,” in *Proc. IEEE ICPADS*, Dec. 2016, pp. 761–768.
[14] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in *Proc. EUROCRYPT*, 2005, pp. 457–473.
[15] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proc. ACM CCS*, 2006, pp. 89–98.
[16] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in *Proc. IEEE S&P*, May 2007, pp. 321–334.
[17] 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.
[18] C. Wang, W. Li, Y. Li, and X. Xu, “A ciphertext-policy attribute-based encryption scheme supporting keyword search function,” in *Cyberspace Safety and Security*. Cham, Switzerland: Springer, 2013, pp. 377–386.

[19] H. Cui, Z. Wan, R. H. Deng, G. Wang, and Y. Li, "Efficient and expressive keyword search over encrypted data in cloud," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 3, pp. 409–422, May/Jun. 2018.
[20] S. Wang, D. Zhao, and Y. Zhang, "Searchable attribute-based encryption scheme with attribute revocation in cloud storage," *PLoS ONE*, vol. 12, no. 8, p. e0183459, 2018.
[21] J. Lai, X. Zhou, R. H. Deng, and K. Chen, "Expressive search on encrypted data," in *Proc. ACM SIGSAC Symp. Inf., Comput. Commun. Secur.*, 2013, pp. 243–252.
[22] Z. Lv, C. Hong, M. Zhang, and D. Feng, "Expressive and secure searchable encryption in the public key setting," in *Proc. Int. Inf. Secur. Conf.*, 2014, pp. 364–376.
[23] T. Peng, L. Qin, B. Hu, J. Liu, and J. Zhu, "Dynamic keyword search with hierarchical attributes in cloud computing," *IEEE Access*, vol. 6, pp. 68948–68960, 2018.
[24] H. Zhu, L. Wang, H. Ahmad, and X. Niu, "Key-policy attribute-based encryption with equality test in cloud computing," *IEEE Access*, vol. 5, pp. 20428–20439, 2018.
[25] E.-J. Goh. (2003). Secure Indexes. IACR ePrint Cryptography Archive. [Online]. Available: http://eprint.iacr.org/2003/216
[26] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proc. ACNS*, 2005, pp. 442–455.
[27] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proc. ACM CCS*, vol. 19, 2006, pp. 79–88.
[28] E. Stefanov, C. Papamanthou, and E. Shi, "Practical dynamic searchable encryption with small leakage," in *Proc. NDSS*, 2014, pp. 1–15.
[29] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 222–233, Jan. 2014.
[30] W. Sun *et al.*, "Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, pp. 3025–3035, Nov. 2014.
[31] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 2, pp. 340–352, Jan. 2016.
[32] H. Li, Y. Yang, T. H. Luan, X. Liang, L. Zhou, and X. S. Shen, "Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 3, pp. 312–325, May/Jun. 2016.
[33] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 9, pp. 2546–2559, Sep. 2016.
[34] Q. Zhang, Q. Liu, and G. Wang, "PRMS: A personalized mobile search over encrypted outsourced data," *IEEE Access*, vol. 6, pp. 31541–31552, 2018.
[35] Y. Zhang, R. H. Deng, J. Shu, K. Yang, and D. Zheng, "TKSE: Trustworthy keyword search over encrypted data with two-side verifiability via blockchain," *IEEE Access*, vol. 6, pp. 31077–31087, 2018.
[36] S. Hu, C. Cai, Q. Wang, X. Luo, and K. Ren, "Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization," in *Proc. IEEE Infocom*, Apr. 2018, pp. 792–800.
[37] H. Yin, Z. Qin, J. Zhang, L. Ou, and K. Li, "Achieving secure, universal, and fine-grained query results verification for secure search scheme over encrypted cloud data," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2017.2709318.
[38] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proc. ACM CCS*, 2007, pp. 456–465.
[39] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography*. Berlin, Germany: Springer, 2011, pp. 53–70.
[40] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proc. EUROCRYPT*, 2011, pp. 547–567.
[41] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proc. ACM CCS*, 2007, pp. 195–203.
[42] M. Chase, "Multi-authority attribute based encryption," in *Proc. TCC*, 2007, pp. 515–534.
[43] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proc. ACM CCS*, 2009, pp. 121–130.
[44] A. Kapadia, P. P. Tsang, and S. W. Smith, "Attribute-based publishing with hidden credentials and hidden policies," in *Proc. NDSS*, 2007, pp. 179–192.
[45] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *Applied Cryptography and Network Security*. Berlin, Germany: Springer, 2008, pp. 111–129.
[46] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: Efficient policy-hiding attribute-based access control," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2130–2145, Jun. 2018.
[47] Y. Zhang, X. Chen, J. Li, D. S. Wong, H. Li, and I. You, "Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing," *Inf. Sci.*, vol. 379, pp. 42–61, Feb. 2017.
[48] J. Camenisch, M. Kohlweiss, A. Rial, and C. Sheedy, "Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data," in *Public Key Cryptography—PKC*. Berlin, Germany: Springer, 2009, pp. 196–214.
[49] W. Zhang, Y. Lin, S. Xiao, J. Wu, and S. Zhou, "Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1566–1577, May 2016.
[50] H. Yin, Z. Qin, L. Ou, and K. Li, "A query privacy-enhanced and secure search scheme over encrypted data in cloud computing," *J. Comput. Syst. Sci.*, vol. 90, pp. 14–27, Dec. 2017.
[51] IETF. *Request for Comments Database*. Accessed: Aug. 28, 2018. [Online] http://www.ietf.org/rfc.html
[52] Hermetic Systems. *Hermetic Word Frequency Counter*. Accessed: 2015. [Online]. Available: http://www.hermetic.ch/wfc/wfc.htm
[53] JPBC. *The Java Pairing Based Cryptography Library*. Accessed: Dec. 4, 2013. [Online]. Available: http://gas.dia.unisa.it/projects/jpbc/index.html

HUI YIN received the B.S. degree in computer science from Hunan Normal University, China, in 2002, the M.S. degree in computer software and theory from Central South University, China, in 2008, and the Ph.D. degree from the College of Information Science and Engineering, Hunan University, China, in 2018. He is currently an Assistant Professor with the College of Applied Mathematics and Computer Engineering, Changsha University, China. His interests include information security, privacy protection, and applied cryptography.



JIXIN ZHANG received the B.S. degree in mathematics from Hubei Polytechnic University and the M.S. degree in computer science and technology from the Wuhan University of Technology, in 2011 and 2014, respectively. He is currently pursuing the Ph.D. degree with the College of Information Science and Engineering, Hunan University, China. His research interests include polymorphic malware detection, privacy protection, applied cryptography, and machine learning.



YINQIAO XIONG received the B.S. and M.S. degrees in computer science and technology from the School of Computer, National University of Defense and Technology, Changsha, China, in 2007 and 2010, respectively, where he is currently pursuing the Ph.D. degree in cyberspace security with the School of Computer. He is currently a Lecturer with Changsha University. His research interests include privacy preserving, information security, and the Internet of Things.

LU OU received the B.S. degree in computer science from the Changsha University of Science and Technology, in 2009, and the M.S. and Ph.D. degrees in software engineering from Hunan University, in 2012 and 2018, respectively. She is a member of the IEEE. Her research interests include security, privacy, optimization, and big data.

FANGMIN LI received the B.Sc. degree from the Huazhong University of Science and Technology, Wuhan, China, in 1990, the M.Sc. degree from the National University of Defense Technology, Changsha, China, in 1997, and the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2001, all in computer science. He is currently a Professor with the School of Computer Engineering and Applied Mathematics, Changsha University. He has authored several books on embedded systems and over 50 academic papers on wireless networks. He holds ten Chinese patents. His current research interests include wireless communications and networks security, computer systems and architectures, and embedded systems. He is a Senior Member of the China Computer Federation (CCF), and a Committee Member of the Technical Committee on Sensor Networks, CCF.

SHAOLIN LIAO received the B.S. in materials science and engineering from Tsinghua University, Beijing, China, in 2000, and the Ph.D. degree in electrical engineering from the University of Wisconsin–Madison, Madison, USA, in 2008. He was a Postdoctoral Fellow with the Department of Physics, The City University of New York, CUNY, from 2008 to 2010. His research interests include the multidisciplinary areas of privacy and machine learning of big data, simulation, algorithms, and modeling in signal processing, as well as novel methods in computational electromagnetics. He is currently a Research and Development Staff with the Argonne National Laboratory and an Adjunct Faculty Member of the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL, USA. He is a Senior Member of the IEEE. He is currently an Associate Editor of the IEEE Access.

KEQIN LI is currently a SUNY Distinguished Professor of computer science with The State University of New York. He has published over 620 journal articles, book chapters, and refereed conference papers. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high-performance computing, CPU–GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, and intelligent and soft computing. He is an IEEE Fellow. He has received several best paper awards. He is currently serving or has served on the Editorial Boards of the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Computers, the IEEE Transactions on Cloud Computing, the IEEE Transactions on Services Computing, and the IEEE Transactions on Sustainable Computing.