Research Article

Fine-Grained Attribute-Based Multikeyword Search for Shared Multiowner in Internet of Things

Yunhong Zhou ¹, Jiehui Nan,² and Licheng Wang ¹

¹State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

²School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, China

Correspondence should be addressed to Licheng Wang; wanglc@bupt.edu.cn

Received 22 December 2020; Revised 4 March 2021; Accepted 1 May 2021; Published 24 May 2021

Academic Editor: Cristina Alcaraz

Copyright © 2021 Yunhong Zhou et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

At present, with the popularity of Internet of things (IoT), a huge number of datasets generated by IoT devices are being uploaded to the cloud storage in remote data management service, but a series of security and privacy defects also arises, where one of the best ways for preventing data disclosure is encryption. Among them, searchable encryption (SE) is considered to be a very attractive cryptographic technology, since it allows users to search records in an encrypted form and to protect user's data on an untrusted server. For the sake of enhancing search permission, attribute-based keyword search (ABKS) is an efficient method to provide secure search queries and fine-grained access authentications over ciphertexts. However, most existing ABKS schemes concentrate on single keyword search, which usually returns redundant and irrelevant results, so it would cost some unnecessary computation and communication resources. Furthermore, existing work in the literature mostly only supports unshared multiowner where a specific data owner owns each file, which is not able to satisfy more desired expressive search. In this work, we propose a novel attribute-based multikeyword search for shared multiowner (ABMKS-SM) primitive in IoT to achieve enhanced access control for users; meanwhile, it can support multikeyword search over ciphertexts and give a formal security analysis in the adaptive against chosen keyword attack (IND-CKA) model. Finally, we have also implemented this prototype to show efficiency when compared with some previous schemes.

1. Introduction

With the rapid advances of Internet of things [1, 2] technology, IoT devices produce large quantities of datasets that require being securely stored and efficiently shared among different users. Such increasing extensive industrial datasets are saved on cloud servers [3] due to large storage capacity, high scalability, and flexible availability. A considerable number of individuals and organizations may be tempted to store their files on the third cloud server, reducing local data storage for convenience. Unfortunately, the cloud storage server is semihonest because it is possible to be curious about the user's stored data in real-world applications [4], and data security concerns have become serious barriers to prevent the widespread usage of cloud storage for IoT. In order to mitigate the concern, the simple and efficient solution is to

encrypt the storage of data to prevent information from being exposed to server, but one limitation is that the encryption mechanism inevitably brings about inflexibilities of data accessing when used to some extent, such as querying keywords on encrypted datasets. In addition, a naive approach is to download whole ciphertexts locally and later on decrypt them for querying, but it leads to wasting computational capabilities and storage costs. So, how to search encrypted keywords securely and efficiently is crucial in an IoT environment.

The positive approach to solve the above problem is using searchable encryption (SE) [5, 6] in which users can not only search directly over encrypted records just like on plaintext space but also preserve the data's privacy. Although SE has paid more attention to the industrial and academic fields for many years, the research is not sufficient because

most data owners wish to share their datasets with legitimate users authorized by them. Furthermore, with the purpose of protecting privacy, the traditional cloud-based access control system is not suitable anymore, because the server cannot be fully trusted. To solve this issue, attribute-based encryption (ABE) [7–14] achieves flexible access authentications over shared data for users, which is a promising cryptographic tool adopted in searchable encryption. Attribute-based keyword search (ABKS) inherits the advantages of SE and ABE, which not only achieves keyword search on encrypted datasets but also preserves the nature of fine-grained access control.

However, for the cloud-based storage system in an IoT environment, the only fine-grained access control is always not adequate. The existing ABKS schemes [15–20] only support single keyword search functions, while it requires massive computational and broadband resources as this retrieval mode returns a lot of irrelevant results. The method to achieve multikeyword search [21–23] was introduced to alleviate the issues. That is to say, when data users use multikeyword search to obtain related records containing multikeyword, the query results enjoy much more desirable accuracy than those using single keyword search. Consequently, how to efficiently construct an attribute-based multikeyword search is significant both in theory and in practice.

More importantly, a practical search system for IoT should support multiple data owners, because a huge number of data files may be shared among different data owners. According to whether a single entity owns each data file or many entities share each other, there are two types of the following multiowner, that is, unshared multiowner [24] and shared multiowner [19, 25]. Previous work in the literature mostly only concentrates on keyword search under unshared multiowner, losing sight of shared multiowner. The shared multiowner setting has many broader and practical applications compared with the unshared multiowner setting, such as cloud-based electronic health record systems. In this system, data records for some patient should be shared by different medical organizations and hospitals. Moreover, the unshared multiowner setting also brings significant computational and storage overheads as each data record is considered to be independent.

Specifically, the authorization privilege of data owners may vary from users to users when considering some practical scenarios in an IoT environment. For example, in a company system, all employees of this company should have the right to search this system. On the one hand, employees in different departments have different search permissions. The attributes of employees can be set; only employees whose attributes meet the access structure specified by a department can query its related records. On the other hand, board members could control the important files of the company in common. The ordinary staff enables decrypting and obtaining the documents with the authorizations of them. The former can use the AND-gate access structure sufficiently to achieve search access control and the latter can use the linear secret-sharing schemes (LSSS) access structure.

In this work, we first put forward an efficient attribute-based multikeyword search for shared multiowner (ABMKS-SM) scheme in Internet of things with fine-grained access control through AND-gate access structure and LSSS. Based on the AND-gate access structure, our proposed ABMKS-SM scheme achieves access control mechanism to enhance the user's search experience, because the AND-gate access structure significantly improves the search algorithm. Based on LSSS technology, our scheme only allows data users to obtain valid authorizations from multiple data owners to decrypt the search results, which is suitable for multiowners sharing scenarios.

In a nutshell, our main contributions can be concluded as follows:

- (1) We first design an efficient and secure ABMKS-SM scheme for IoT environments, where multiple data owners enable controlling user's search permissions and only legitimate users with authorizations could search for outsourced data. More importantly, it can be applied to shared multiowner settings.
- (2) The most important security goal of designing schemes is adaptive indistinguishable security against chosen keyword attack (IND-CKA). We present a formal security analysis in an IND-CKA model while guaranteeing the privacy of keywords.
- (3) In the respect of functional comparison, theoretical analysis, and experimental analysis, we evaluate our scheme's performance and further demonstrate the efficiency and practicality of this scheme. At last, from theoretical analysis, it is shown that our ABMKS-SM scheme is superior to the previous CP-ABKS [15] and ABKS-SM scheme [19]. From experimental analysis, the obtained results further demonstrate computation costs to search which are free from attributes; meanwhile, the time costs to generate the trapdoor are not related to the number of attributes.

The remainder of this paper can be organized as follows. We recall some related work in Section 2. We describe some necessary cryptographic tools in Section 3. We give the system model, scheme definition, and security model in Section 4. We propose a new and concrete scheme in Section 5. We discuss a formal security analysis and performance analysis when compared with previous schemes in Section 6. We conclude this paper in Section 7.

2. Related Work

In 2000, Song et al. [5] suggested symmetric searchable encryption (SSE) scheme by using symmetric cryptography, which first introduces the concept of searchable encryption. Subsequently, Boneh et al. [26] seminally designed searchable public key encryption construction exploring identity-based encryption for e-mail systems in an asymmetric setting, where the data owner extracts keywords from messages and encrypts them before outsourcing to the server, then the data user can generate his interested

keyword search token under his private key, and then he sends this token to the server. Upon receipt of the user's token, the server can perform the keyword retrieval operations and returns the related search results. Since this seminal work, many researchers have made great efforts [27–35] and proposed a series of searchable encryption proposals to make them more efficient and enrich the search diversified, for example, fuzzy search [36–38], conjunctive search [39–41], and ranked search [42–44].

To the best of our knowledge, ABE [7] implements attribute-based access control as an efficient solution. Furthermore, ABKS schemes can search keywords over encrypted documents with access control by utilizing ABE technology. In 2014, Zheng et al. [15] developed an attribute-based keyword search primitive and provided two concrete ABKS scheme constructions. In an ABKS scheme, data owners encrypt keywords and build searchable indexes embedded in an access structure, and only legitimate users can generate their search tokens for querying on outsourced datasets. Later on, Dong et al. [16] provided an efficient ABKS construction for resource-constrained mobile devices via an online/offline approach, where data owners and users are allowed to execute the related algorithm in this way. More specifically, an outsourcing key-issuing and decryption scheme was designed by Li et al. [17], where the cloud server can decrypt partial work without learning anything about the message. Recently, Qiu et al. [18] provided an enhanced scheme to achieve an access policy with hidden, where the data owner implements fine-grained authorizations for different users with a hidden structure in encrypted form. However, all the aforementioned schemes only consider searching a single keyword.

Zhang et al. [21] introduced a searchable design with ranked multikeyword under multiple owners setting where the security of keywords and documents could be protected. Accordingly, by exploring proxy re-encryption as well as lazy re-encryption technology, an authorized keyword search construction was designed by Sun et al. [22] under multiple data owners and users. And it can achieve an efficient user revocation mechanism. Subsequently, Miao et al. [24] provided a new multikeyword search proposal on medical records that is encrypted via ciphertext-policy attribute-based encryption. In their construction, it not only offers multikeyword search but also can be applied to multiowner settings. On the other hand, Liu et al. [23] presented a new improved scheme with user tracing using the AND-gate on the multivalued attribute. Although these above schemes support multikeyword search, it cannot be applied to scenarios where multiple owners could be shared. More recently, Miao et al. [19] first presented an efficient ABKS scheme which is suitable for shared multiowner setting; however, it only considers a single keyword search. Moreover, Miao et al. [25] suggested a scheme to deal with conjunctive keyword search with verification on the basis of multisignatures in shared multiowner setting again. Although their scheme is applied to a shared multiowner setting, it also fails to support multikeyword search. There are no efficient attribute-based multikeyword search schemes for shared multiowner so far. Thus, in this work, we

first design an efficient attribute-based multikeyword search for shared multiowner (ABMKS-SM) scheme in IoT, and it can support the multikeyword search on ciphertexts with fine-grained access control.

3. Preliminaries

We explain some necessary cryptographic tools related to our ABMKS-SM construction.

3.1. Bilinear Map. Suppose $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of same order p , and g is a generator of $\mathbb{G}$. Let $\hat{e}: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ represent a computable bilinear map that satisfies three conditions as below:

- (1) Bilinearity: for any $g, h \in \mathbb{G}$ and $x, y \in \mathbb{Z}_p^*$, there exists $\hat{e}(g^x, h^y) = \hat{e}(g, h)^{xy}$
- (2) Nondegeneracy: for $g \in \mathbb{G}$, $\hat{e}(g, g) \neq 1$
- (3) Computability: given $g, h \in \mathbb{G}$, the pairing $\hat{e}(g, h)$ could be computed efficiently

3.2. Access Structure. In our scheme, we define an AND-gate access structure [45, 46] based multivalued attribute. Suppose $L = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$ denotes an attribute list where n represents the number of attributes. Each attribute $\lambda_i \in L$, ($i = 1, 2, \dots, n$) has a possible value set $V_i = \{v_{i,1}, v_{i,2}, \dots, v_{i,n_i}\}$, where n_i is the number of possible values of λ_i . The user's attribute set S_U is defined as $S_U = \{a_1, a_2, \dots, a_n\}$, where $a_i \in V_i$. The AND-gate access policy is denoted as $\mathbb{P} = \{\text{att}_1, \text{att}_2, \dots, \text{att}_n\}$, where $\text{att}_i \in V_i$. If $a_i = \text{att}_i$ ($i \in [1, n]$), the attribute set for users S_U satisfies the AND-gate access policy $\mathbb{P}$. Specifically, the user's attribute list is the same structure as the access policy when the attribute in the user's attribute list has only one value.

3.3. Linear Secret-Sharing Schemes (LSSS). Linear secret-sharing schemes (LSSS) [47] can convert previously used structures such as formulas (equivalently tree structures) into an LSSS representation by using standard techniques [48] and enhance the access control to multiparty requirements. Suppose $P = \{P_1, P_2, \dots, P_l\}$ is a collection of parties, a secret-sharing scheme Π is called linear (over $\mathbb{Z}_p$) on the condition that the following properties are satisfied.

- (i) A vector over $\mathbb{Z}_p$ is formed by the shares for each party P_i ($i \in [1, l]$).
- (ii) There exists a linear secret-sharing structure (M, ρ) , where $M_{l \times n}$ denotes a sharing-generating matrix and a monotone function $\rho(i)$ ($i \in [1, l]$) can label the i -th row in M , where ρ denotes an injective function from $\{1, 2, \dots, l\}$ to a party. Given random elements $r_2, \dots, r_n \in \mathbb{Z}_p^*$, we consider constructing a column vector $\vec{v} = (s, r_2, \dots, r_n)$ and compute the shares of secret s as $\lambda_i = M_i \vec{v}$, where the share λ_i belongs to P_i and M_i represents the i -th row in M .

According to the above definition, every linear secret-sharing scheme satisfies linear reconstruction property.

Assume Π is an LSSS for the access structure $\mathbb{A}$ and $\mathcal{S}$ represents an authorized set that satisfies $\mathbb{A}$ (namely, $\mathcal{S} \in \mathbb{A}$). We define $I \subset \{1, 2, \dots, l\}$ as $I = \{i | \rho(i) \in \mathcal{S}\}$. If $\{\lambda_i\}$ are the valid shares of a secret s , then we can find a constant set $\{\omega_i\}_{i \in I}$ such that $\sum_{i \in I} \omega_i M_i = (1, \dots, 0)$ via Gaussian elimination method. Consequently, the equation $\sum_{i \in I} \omega_i \lambda_i = s$ can be satisfied.

3.4. Decisional Bilinear Diffie-Hellman Assumption.

Decisional Bilinear Diffie-Hellman (DBDH) Assumption has the following definition: given elements $g^x, g^y, g^z \in \mathbb{G}$ where $x, y, z \in \mathbb{Z}_p^*$ and g is a generator of group $\mathbb{G}$, the DBDH problem is to distinguish $\hat{e}(g, g)^{xyz} \in \mathbb{G}_T$ from a random group element $\hat{e}(g, g)^\gamma \in \mathbb{G}_T$ where $\gamma \in \mathbb{Z}_p^*$. It is said that the DBDH assumption holds if there no exists a probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$ who has a nonnegligible advantage solving DBDH hardness problem, where the advantage function of this algorithm $\mathcal{A}$ can be denoted as

$$\text{Adv}_{\mathcal{A}}^{\text{DBDH}}(\lambda) = |\Pr[\mathcal{A}(g, g^x, g^y, g^z, \hat{e}(g, g)^{xyz}) = 1] - \Pr[\mathcal{A}(g, g^x, g^y, g^z, \hat{e}(g, g)^\gamma) = 1]|. \quad (1)$$

4. System and Security Model

We give a description of the system model, scheme definition and corresponding security model, respectively.

4.1. System Model. We discuss our ABMKS-SM system for shared multiowner settings in IoT, which consists of four participants, including cloud service provider (CSP), data users, trusted authority (TA) and multiple data owners. TA initializes the system and distributes keys for multiple data owners and users. First, multiple data owners encrypt files with symmetric keys and further encrypt symmetric keys with a random secret value. In particular, based on LSSS access structure, multiple data owners share the secret value with each other when considering shared multiowner settings. Then, multiple data owners build keyword indexes extracted keywords from each document under the AND-gate access policy before outsourcing to the CSP. Especially, the CSP provides computation, storage and search services for users. When a user wishes to request a keyword query on storage of encrypted records, he could produce a search keyword trapdoor and then submit it into the CSP. Having received a trapdoor from a data user, the CSP attempts to retrieve over encrypted data and returns relevant results to users. Finally, the user decrypts corresponding results only if he gets legitimate authorization credentials associated with multiple data owners. More specifically, our system model is presented in Figure 1.

4.2. Algorithm Definition in ABMKS-SM System. In this section, we give the algorithm defined in the ABMKS-SM system comprised the following six algorithms.

Setup(1^λ): TA runs the setup algorithm, which inputs the security parameter λ , and generates master key msk and public parameter pp .

KeyGen($\text{pp}, \mathcal{O}, \text{ID}, S_U, \text{msk}$): TA runs this key generation algorithm, which inputs the public parameter pp , the multiple data owners set $\mathcal{O}$, the user's identity ID , the data user's attribute set S_U and the master key msk , and generates public key $\text{PK}_{\mathcal{O}_j}$ and secret key $\text{SK}_{\mathcal{O}_j}$ of each data owner $\mathcal{O}_j$ and user's private key SK_U .

Encryption($W, \text{pp}, (F, K), \mathbb{P}, \text{PK}_{\mathcal{O}}, (M, \rho)$): the multiple data owners run the encryption algorithm, which inputs the keyword set W , the public parameter pp , the file/symmetric key pair set $(F, K) = \{f, k\}$, the AND-gate access policy $\mathbb{P}$ used to construct keyword indexes, the public key $\text{PK}_{\mathcal{O}}$ of data owners and an access policy (M, ρ) to multiple data owners authorizations for accredited data users, and generates ciphertexts CT and the encrypted indexes I_W .

Trapdoor($\text{pp}, S_U, \text{SK}_U, W'$): the data user runs this trapdoor generation algorithm, which inputs the public parameter pp , the attribute set S_U , the private key SK_U and a set of the query keyword W' , and generates the search trapdoor $T_{W'}$.

Search($\text{pp}, T_{W'}, S_U, I_W$): the CSP runs the search query algorithm, which inputs the public parameter pp , the trapdoor $T_{W'}$, the attribute set S_U and the encrypted index I_W . When S_U satisfies AND-gate access policy $\mathbb{P}$ contained I_W , and further the search trapdoor and the encrypted index contain the same keyword set, this algorithm returns 1 and sends the relevant search results CT' to user. Otherwise, it returns 0.

Decryption($\text{pp}, \text{SK}_U, \text{ID}, \text{CT}'$): the data user runs this decryption algorithm, which inputs the public parameter pp , private key SK_U , user's identity ID and the corresponding search results CT' . If the user's identity ID is authorized by multiple data owners, it decrypts relevant search results.

4.3. Security Model. Suppose $\mathcal{A}$ is an adversary and $\mathcal{C}$ is a challenger, we define our ABMKS-SM scheme's security model as adaptive indistinguishable security against chosen keyword attack (IND-CKA) game, which is conducted between $\mathcal{A}$ and $\mathcal{C}$ as the following steps.

Setup: $\mathcal{C}$ runs the *Setup*(1^λ) algorithm to generate pp and msk . $\mathcal{C}$ runs the *KeyGen*($\text{pp}, \mathcal{O}, \text{ID}, S_U, \text{msk}$) to output SK_U , while $\mathcal{A}$ only owns public parameters pp .

Phase 1: $\mathcal{A}$ is able to adaptively request the trapdoor from a tuple of keyword sets W' . $\mathcal{C}$ executes the *Trapdoor*($\text{pp}, S_U, \text{SK}_U, W'$) algorithm and generates the search trapdoor $T_{W'}$, and then returns it to $\mathcal{A}$.

FIGURE 1: ABMKS-SM model.

Challenge: $\mathcal{A}$ chooses keyword sets W_0, W_1 for challenging, and submits to $\mathcal{E}$. It is required that the two keyword sets are not able to query in *Phase 1*. Upon receipt of two keyword sets, $\mathcal{E}$ selects a random bit $b \in \{0, 1\}$ to output an encrypted index I_{W_b} , and sends it to the adversary $\mathcal{A}$.

Phase 2: $\mathcal{A}$ can request the queries for more trapdoors of keyword sets and the only restriction is that any keyword set of his choice except for the W_0, W_1 .

Guess: finally, $\mathcal{A}$ outputs a guess b' , if $b = b'$, $\mathcal{A}$ gains this game.

We define that $\mathcal{A}$'s the advantage function against IND-CKA game can be denoted as

$$\text{Adv}_{\mathcal{A}}^{\text{IND-CKA}}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|. \quad (2)$$

Definition 1. A privacy-preserving ABMKS-SM scheme is IND-CKA secure under the circumstance that the advantage of breaking IND-CKA game is negligible for any PPT adversary $\mathcal{A}$.

5. ABMKS-SM Construction

Based on the algorithm defined in the ABMKS-SM system, we present a specific construction of our proposed scheme that makes use of six algorithms. The running algorithms are described below.

Setup (1^λ): given the security parameter λ , it generates pp and msk , which works as follows:

TA first randomly selects two cyclic groups, $\mathbb{G}$ and $\mathbb{G}_T$, with same order p and sets a computable bilinear pairing $\hat{e}: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ and g is the generator of $\mathbb{G}$.

Then, the TA selects two secure hash functions: $H_1: \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$ and $H_2: \{0, 1\}^* \rightarrow \mathbb{G}$. Finally, TA randomly selects $\alpha, \beta \in \mathbb{Z}_p^*$, computes $g^\beta, A = \hat{e}(g, g)^\alpha$, and sets $pp = (G, G_T, p, \hat{e}, g, A, g^\beta, H_1, H_2), msk = (\alpha, \beta)$

KeyGen($pp, \mathcal{O}, ID, S_U, msk$): given the public parameter pp , the multiple data owners set $\mathcal{O} = \{\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_l\}$, the user's identity ID , the attribute set S_U , and the master key msk , it generates the public key $PK_{\mathcal{O}_j}$ and secret key $SK_{\mathcal{O}_j}$ of each data owner ($\mathcal{O}_j, j \in [1, l]$) and the user's private key SK_U , which works as follows:

TA randomly chooses $x_j \in \mathbb{Z}_p^*$ and sets each data owner's public key and secret key as $PK_{\mathcal{O}_j} = g^{x_j}, SK_{\mathcal{O}_j} = x_j$. The TA randomly chooses $r \in \mathbb{Z}_p^*$, computes $K_1 = g^r, K_2 = g^\alpha$, and sets $K_3 = K_2 H_2(ID)^\beta$. For each attribute $a \in S_U$, compute $K_a = K_1 g^{H_1(a)}$. Set the data user's private key as $SK_U = (K_1, K_2, K_3, \{(K_a) | \forall a \in S_U\})$

Encryption($W, pp, (F, K), \mathbb{P}, PK_{\mathcal{O}}, (M, \rho)$): given the keyword set $W = \{w_1, w_2, \dots, w_m\}$, the public parameter pp , the file/symmetric key pair set $(F, K) = \{f, k\}$, the AND-gate access policy $\mathbb{P}$ used to establish encrypted keyword indexes, the data owner's public key $PK_{\mathcal{O}}$, and an LSSS access policy (M, ρ) to multiple data owners authorizations for accredited data users, where $M_{l \times n}$ is an access matrix and ρ is a function mapping each row of M to a data owner, this encryption algorithm generates ciphertexts CT and the encrypted indexes I_W .

- (i) The ciphertexts CT are generated as follows:
For each file f , one of multiple data owners (without loss of generality, assume the data owner

is $\mathcal{O}_1$) encrypts f with symmetric key $k \in \mathbb{G}_T$ as c , that is, $f \xrightarrow{k} c$.

Then, $\mathcal{O}_1$ encrypts each symmetric key k and randomly chooses the secret value s and computes $C' = kA^s, C'' = g^s$.

After, $\mathcal{O}_1$ randomly selects values $r_2, \dots, r_n \in \mathbb{Z}_p^*$ and considers to build a column vector $\vec{v} = (s, r_2, \dots, r_n)$ and further computes the shares of the secret s as $\lambda_j = M_j \cdot \vec{v} \ (j \in [1, l])$. $\mathcal{O}_1$ sends remaining λ_j to other data owners via a secure channel. Data owners compute $C_j = g^{\beta \lambda_j} \text{PK}_{\mathcal{O}_j}^{-1} = g^{\beta \lambda_j} g^{-x_j}, j \in [1, l]$.

Set symmetric key ciphertexts as

$$C = (C', C'', \{C_j\}). \quad (5)$$

The ciphertexts are

$$\text{CT} = (c, C) = (c, C', C'', \{C_j\}). \quad (6)$$

- (ii) The encrypted indexes I_W are generated as follows: Multiple data owners extract keyword sets $W = \{w_1, w_2, \dots, w_m\}$ from the file set $F = \{f\}$ and select a random element $\theta \in \mathbb{Z}_p^*$. For any keyword $w_i \in W$, multiple data owners compute $I_{w_i} = A^\theta \hat{e}(g^\theta, H_2(w_i))$. Compute $I_1 = g^\theta$. For each attribute $\text{att} \in \mathbb{P}$, compute $I_{\text{att}} = g^{\theta H_1(\text{att})}$. Set the encrypted indexes as $I_w = (I_w, I_1, \{I_{\text{att}}\})$.

Besides, multiple data owners have authorized users' identity list, and each data owner can generate the valid decryption authorization $\{A_{\mathcal{O}_j} = H_2(\text{ID})^{x_j}\}$ with his/her private key $\text{SK}_{\mathcal{O}_j}$ for the identity ID of a data user.

Trapdoor (pp, S_U, SK_U, W'): given the public parameter pp, the attribute set S_U , the private key SK_U , and the search keyword set $W' = \{w'_1, w'_2, \dots, w'_t\}$, this trapdoor generation algorithm generates the search trapdoor $T_{W'}$, which works as follows:

The data user calculates $T = K_1 K_2^t \prod_{i=1}^t H_2(w'_i) = g^r g^{\text{att}} \prod_{i=1}^t H_2(w'_i)$.

For each attribute $a \in S_U$, compute $T_a = K_a = g^r g^{H_1(a)}$.

Set the trapdoor as $T_{W'} = (T, \{T_a\} \mid \forall a \in S_U)$

Search (pp, $T_{W'}, S_U, I_W$): given the public parameter pp, the query trapdoor $T_{W'}$, the attribute set S_U , and the encrypted indexes I_W , the search query algorithm returns 1 or 0, which works as follows.

If S_U satisfies the AND-gate access policy $\mathbb{P}$ embedded in I_W , then the CSP checks that the following equation holds:

$$\hat{e}(I_1, T_a) \prod_{i=1}^t I_{w_i} = \hat{e}(I_1, T) \hat{e}(g, I_{\text{att}}). \quad (9)$$

If the above condition holds, this search algorithm returns 1 and sends the relevant search results CT' to user; otherwise, it returns 0.

Decryption (pp, $\text{SK}_U, \text{ID}, \text{CT}'$): given the public parameter pp, the user's private key SK_U and identity ID, and the relevant query results CT' , the decryption algorithm returns related file encryption key set $K = \{k\}$, which works as follows.

The data user first verifies whether the identity ID is authorized from multiple data owners. Note that there is no intersection between data users and data owners. If it is not in the authorized users' identity list, the algorithm returns 0.

Otherwise, the identity ID of the data user obtains the valid decrypted authorizations $\{A_{\mathcal{O}_j} = H_2(\text{ID})^{x_j}\}$ from multiple data owners.

Assume $\mathbb{A}$ is an LSSS matrix access policy (namely, $\mathbb{A} = (M, \rho)$) and $\mathcal{S}$ is an authorized set of data owners ($\mathcal{S} \in \mathbb{A}$) with $\eta = \{j \mid \rho(j) \in \mathcal{S}\} \subset \{1, 2, \dots, l\}$. When multiple data owners encrypted each shared file with same access structure, they get a constant set $\{\varphi_j\}_{j \in \eta}$ by solving the equation $\sum_{j \in \eta} \varphi_j M_i = (1, \dots, 0)$.

According to the decrypted authorizations $\{A_{\mathcal{O}_j}\}_{\mathcal{O}_j \in \mathcal{S}}$ and the private key SK_U , the data user computes the following equation:

$$\frac{\hat{e}(C'', K_3)}{\prod_{j \in \eta} \left(\hat{e}(C_j, H_2(\text{ID})) \hat{e}(A_{\mathcal{O}_j}, g) \right)^{\varphi_j}} = \hat{e}(g, g)^{\text{as}}. \quad (10)$$

Finally, the data user gains the file encryption key $k = (C' / \hat{e}(g, g)^{\text{as}})$ and decrypts the related search results.

6. Security and Performance

We analyze our scheme's correctness, security, and performance.

6.1. Correctness. If S_U satisfies the AND-gate access policy $\mathbb{P}$ embedded in I_W , while the search trapdoor and the encrypted index contain same keyword set, $W' \subseteq W$, we can verify the correctness of search algorithm indicated as

$$\begin{aligned} & \hat{e}(I_1, T_a) \prod_{i=1}^t I_{w_i} \\ &= \hat{e}(g^\theta, g^r g^{H_1(a)}) \prod_{i=1}^t \hat{e}(g, g)^{\text{att}} \hat{e}(g^\theta, H_2(w_i)) \\ &= \hat{e}(g, g)^{\text{att}} \prod_{i=1}^t \hat{e}(g^\theta, H_2(w_i)) \hat{e}(g^\theta, g^r) \hat{e}(g^\theta, g^{H_1(a)}) \\ &= \hat{e}(g, g)^{\text{att}} \hat{e}\left(g^\theta, \prod_{i=1}^t H_2(w_i)\right) \hat{e}(g^\theta, g^r) \hat{e}(g^\theta, g^{H_1(a)}) \\ &= \hat{e}(g, g)^{\text{att}} \hat{e}\left(g^\theta, \prod_{i=1}^t H_2(w_i)\right) \hat{e}(g, g)^{\theta r} \hat{e}(g, g)^{\theta H_1(a)}, \end{aligned} \quad (11)$$

where the encryption algorithm generates I_{w_i} and I_1 and the trapdoor algorithm generates T_a .

$$\begin{aligned}
& \widehat{e}(I_1, T) \widehat{e}(g, I_{\text{att}}) \\
&= \widehat{e}\left(g^\theta, g^r g^{\alpha t} \prod_{i=1}^t H_2(w_i')\right) \widehat{e}(g, g^{\theta H_1(\text{att})}) \\
&= \widehat{e}(g^\theta, g^r) \widehat{e}(g^\theta, g^{\alpha t}) \widehat{e}\left(g^\theta, \prod_{i=1}^t H_2(w_i')\right) \widehat{e}(g, g^{\theta H_1(\text{att})}) \\
&= \widehat{e}(g, g)^{\theta r} \widehat{e}(g, g)^{\theta \alpha t} \widehat{e}\left(g^\theta, \prod_{i=1}^t H_2(w_i')\right) \widehat{e}(g, g)^{\theta H_1(\text{att})},
\end{aligned} \tag{12}$$

where the encryption algorithm generates I_1 and I_{att} and the trapdoor algorithm generates T .

If the identity ID of the data user is authorized from multiple data owners, according to the decrypted authorizations $\{A_{\theta_j}\}_{\theta_j \in \mathcal{S}}$ and the private key SK_U , then we can verify correctness for decryption indicated as

$$\begin{aligned}
& \frac{\widehat{e}(C'', K_3)}{\prod_{j \in \eta} \left(\widehat{e}(C_j, H_2(\text{ID})) \widehat{e}(A_{\theta_j}, g) \right)^{\varphi_j}} \\
&= \frac{\widehat{e}(g^s, g^\alpha H_2(\text{ID})^\beta)}{\prod_{j \in \eta} \left(\widehat{e}(g^{\beta \lambda_j} g^{-x_j}, H_2(\text{ID})) \widehat{e}(H_2(\text{ID})^{x_j}, g) \right)^{\varphi_j}} \\
&= \frac{\widehat{e}(g^s, g^\alpha) \widehat{e}(g^s, H_2(\text{ID})^\beta)}{\widehat{e}(H_2(\text{ID})^{x_j}, g) \widehat{e}(g^{-x_j}, H_2(\text{ID})) \widehat{e}(g^\beta, H_2(\text{ID}))^{\sum_{j \in \eta} \varphi_j \lambda_j}} \\
&= \frac{\widehat{e}(g^s, g^\alpha) \widehat{e}(g^s, H_2(\text{ID})^\beta)}{\widehat{e}(g^\beta, H_2(\text{ID}))^s} \\
&= \widehat{e}(g, g)^{\alpha s}.
\end{aligned} \tag{13}$$

6.2. Security Proof. In the area of public key searchable encryption, IND-CKA secure is one of the most important security goals. By the described above security model defined in Section 4.3, we formally prove the security of ABMSK-SM construction.

Theorem 1. *Our ABMSK-SM scheme is IND-CKA secure provided that the DBDH assumption holds.*

Proof 1. In fact, our reduction is straightforward. Intuitively, assume that there exists an adversary $\mathcal{A}$ that can break our proposed scheme. We could build a simulator $\mathcal{B}$ who resolves DBDH problems of distinguishing the DBDH tuple $(g^x, g^y, g^z, Z = \widehat{e}(g, g)^{xyz})$ and a random tuple

$(g^x, g^y, g^z, Z = \widehat{e}(g, g)^y)$, where $x, y, z, \gamma \in \mathbb{Z}_p^*$. Next, we formally show the following reduction.

Init: at first, $\mathcal{A}$ chooses an AND-gate access policy $\mathbb{P}^*$ for challenging and returns to $\mathcal{B}$.

Setup: $\mathcal{B}$ selects random elements $\alpha, \beta \in \mathbb{Z}_p^*$ and calculates $A = \widehat{e}(g, g)^\alpha, g^\beta$. $\mathcal{B}$ selects one hash function $H_1: \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$ and sets $\text{msk} = (\alpha, \beta)$ and $\text{pp} = (\mathbb{G}, \mathbb{G}_T, p, \widehat{e}, g, A, g^\beta)$.

The hash query $H_2(w_i)$ is simulated as a random oracle model as below. If w_i has not been requested previously, the simulator $\mathcal{B}$ randomly chooses $\rho_i \in \mathbb{Z}_p^*$ and next puts (w_i, ρ_i) to the list O_{H_2} and outputs g^{ρ_i} ; otherwise, the simulator $\mathcal{B}$ searches ρ_i from O_{H_2} and returns g^{ρ_i} .

Phase 1: $\mathcal{A}$ can query the trapdoor for keyword set W and request the queries of the following O_{KeyGen} and O_{Trapdoor} oracles.

- (1) O_{KeyGen} : the simulator $\mathcal{B}$ randomly selects $r \in \mathbb{Z}_p^*$ and computes $K_1 = g^r$. For each attribute $a \in S_U$, compute $K_a = K_1 g^{H_1(a)} = g^r g^{H_1(a)}$. $\mathcal{B}$ returns SK_U to $\mathcal{A}$ and stores a private key's list L_{SK_U} .
- (2) O_{Trapdoor} : at the beginning, the simulator $\mathcal{B}$ issues the O_{KeyGen} oracle to gain the secret key SK_U and then $\mathcal{B}$ calculates $T = g^r g^\alpha \prod_{i=1}^t g^{\rho_i}$ and $T_a = g^r g^{H_1(a)}$ for each attribute $a \in S_U$. $\mathcal{B}$ adds W to the list L_W depending on S_U which satisfies the access policy.

Challenge: $\mathcal{A}$ submits $\mathcal{B}$ two keyword sets W_0, W_1 . Without loss of generality, W_0 and W_1 are not in L_W . $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$ and encrypts W_b to generate encrypted keyword index I_{W_b} . It is shown as follows: $\mathcal{B}$ computes $I_{w_i}^* = Z \widehat{e}(g^x, H_2(w_i))$ for any keyword $w_i \in W_b$, $I_1^* = g^x$, and $I_{\text{att}} = g^{\theta H_1(\text{att})}$ for each attribute att in the access policy $\mathbb{P}^*$. Finally, $\mathcal{B}$ sends the encrypted keyword index $I_{W_b} = (I_{w_i}^*, I_1^*, \{I_{\text{att}}\})$ to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ can repeat the queries of more trapdoors for keyword sets; notice that any keyword set is of his choice except for the W_0, W_1 .

Guess: finally, $\mathcal{A}$ outputs b' . $\mathcal{B}$ returns 1 if $b = b'$; otherwise, it randomly returns.

It has the following two conditions:

- (i) If $Z = \widehat{e}(g, g)^{xyz}$, $\mathcal{A}$ is given a ciphertext I_{W_b} , and we suppose that $\mathcal{A}$ wins this game with an advantage ε .
 $I_{w_i}^* = Z \widehat{e}(g^\theta, H_2(w_i)) = \widehat{e}(g, g)^{xyz} \widehat{e}(g^x, H_2(w_i))$
 $(w_i \in W_b) I_1^* = g^x$, whereas α, θ are random elements, $x, y, z \in \mathbb{Z}_p^*$, let $x = \theta, yz = \alpha$, $I_{w_i}^* = A^\theta \widehat{e}(g^\theta, H_2(w_i)), I_1^* = g^\theta$, which means I_{W_b} is a valid ciphertext.

Since $\mathcal{A}$ has an advantage ε with its correct guess, we make a conclusion that $\Pr[\mathcal{B}(g, g^x, g^y, g^z, Z = \widehat{e}(g, g)^{xyz}) = 1] = (1/2) + \varepsilon$.

- (ii) Otherwise, I_{W_b} is a random ciphertext. $\mathcal{A}$ is not able to obtain any advantage in breaking IND-CKA game, so that we have $\Pr[\mathcal{B}(g, g^x, g^y, g^z, Z = \hat{e}(g, g)^y) = 1] = (1/2)$.

Therefore, in the IND-CKA security game, the overall advantages of $\mathcal{B}$ solving DBDH problems can be denoted as

$$\text{Adv}_{\mathcal{B}}^{\text{IND-CKA}}(\lambda) = \frac{1}{2} \left(\frac{1}{2} + \varepsilon \right) + \frac{1}{2} \cdot \frac{1}{2} - \frac{1}{2} = \frac{\varepsilon}{2}. \quad (14)$$

In other words, the advantage of a simulator $\mathcal{B}$ solving DBDH problems is negligible because the advantage of a PPT adversary $\mathcal{A}$ against the IND-CKA security game is negligible. It can be said that our ABMKS-SM scheme is IND-CKA secure provided that the DBDH assumption holds. This proves the security of our scheme. $\square$

6.3. Performance Analysis. From the aspect of functional comparison, theoretical analysis, and experimental analysis, we show our performance.

6.3.1. Functional Comparison. In terms of functionalities, mainly including attribute-based keyword search, multi-keyword search, and unshared multiowner as well as shared multiowner, we compared our proposed scheme with some previous schemes, as demonstrated in Table 1. One observes that our ABMKS-SM scheme has much richer capabilities that can support all the above types of functionalities at the same time, which enables our scheme to be used in IoT.

6.3.2. Theoretical Analysis. We analyze computation and storage costs in terms of theoretical analysis. At first, we introduce several time-consuming operations, such as hash operation H mapping to the element in group $\mathbb{G}$, pairing operation Pair , multiplication operation M_T in group $\mathbb{G}_T$, and modular exponentiation operation E (or E_T) in group $\mathbb{G}$ (or $\mathbb{G}_T$). For ease of comparison, we ignore multiplication operation in group $\mathbb{G}$ as well as hash operation which maps to an element in $\mathbb{Z}_p^*$. As shown in Table 2, we give detailed notation definitions of the performance analysis.

To better assess the efficiency of our proposed ABMKS-SM scheme, we make a comparison of state-of-the-art CP-ABKS [15], ABKS-SM [19], and our ABMKS-SM scheme. Table 3 shows the computation costs of compared schemes. We take into account the computation costs by evaluating *KeyGen*, *Encryption*, *Trapdoor*, and *Search* algorithm. From Table 3, it is worth noticing that our construction has much more efficiency than other schemes, especially for *KeyGen*, *Trapdoor*, and *Search* algorithm. In *KeyGen* algorithm, our scheme just needs $(n + l + 3)E + H$ time, but the CP-ABKS and ABKS-SM scheme take $(2n + 2)E + nH$ and $(2n + l + 4)E + E_T + H$ time, respectively. So, our scheme outperforms ABKS-SM and CP-ABKS scheme regarding the time of key generation. In *Trapdoor* algorithm, the computation

costs of the ABKS-SM and CP-ABKS scheme increase linearly along with the number of attributes, while our ABMKS-SM construction almost remains unchanged and only takes $E + tH$ time. Our time costs are related to the number of search keywords, but hash operation H is less than exponentiation operation E . Therefore, our construction is more superior to CP-ABKS and ABKS-SM scheme regarding generating the trapdoor time. In *Search* algorithm, our scheme just needs $3\text{Pair} + (t + 2)M_T$ time, while the time of CP-ABKS and ABKS-SM scheme is subject to the number of system attributes, so our scheme offers a much better search experience. In *Encryption* and *Search* algorithm, the number of keywords also influences the time of our scheme, but it does not reduce the search experience for data users and can support more rich functionalities; therefore, our scheme is still desirable in the Internet of things environment.

As shown in Table 4, we compare storage costs by evaluating *KeyGen* algorithm, *Encryption* algorithm, and *Trapdoor* algorithm. From Table 4, one observed that the storage costs of *Trapdoor* algorithm of our ABMKS-SM scheme outperform the CP-ABKS [15] and ABKS-SM [19] scheme. Along with the number of attributes growth increased in *KeyGen* algorithm, the storage costs of the ABKS-SM scheme and our ABMKS-SM scheme show an upward trend as a result of supporting shared multiple data owners. More concretely, our ABMKS-SM scheme achieves higher efficiency than the ABKS-SM scheme. In *Encryption* algorithm, our scheme's storage costs are on the rise with the number of keywords in the encryption phase because of supporting multikeyword search. As we all know, supporting more complex functionalities can sometimes sacrifice some efficiency, but it does not bring a great influence on user search experience. Accordingly, our ABMKS-SM scheme can be accepted for more practical applications.

6.3.3. Experimental Analysis. To validate the theoretical analysis, we implement our scheme in software by using JPBC library [49] in JRE1.8 environment. Furthermore, we simulate our experiments on the Windows 10 system with a laptop with Intel(R) Core (TM) i7-8565U CPU, 8.00 GB RAM through using Java language. In order to achieve a practical function, we choose an elliptic curve group with Type A: $y^2 = x^3 + x$. For ease of description, we mainly take into account several phases, such as key generation, encryption, trapdoor generation, search, and decryption process. Further, we evaluate our performance by varying the number of keywords and attributes and set $m = t$.

As illustrated in Figure 2, when the number of keywords is 10, we can see the time costs of key generation, encryption, trapdoor generation, and search process change as the number of attributes grows larger. In this case, it is shown that the computation costs of trapdoor generation, decryption, and search process have obvious advantages, which are free from the number of data user's attributes. Moreover,

TABLE 1: Functional comparison.

Scheme	Attribute-based keyword search	Multikeyword search	Unshared multiowner	Shared multiowner
[15]	✓			
[18]	✓			
[19]	✓		✓	✓
[20]	✓			
[21]		✓	✓	
[22]	✓	✓	✓	
[23]	✓	✓		
[24]	✓	✓	✓	
[25]		✓	✓	✓
Ours	✓	✓	✓	✓

TABLE 2: Notation definitions for performance analysis.

Notation	Description
E	A modular exponentiation operation time in $\mathbb{G}$
E_T	A modular exponentiation operation time in $\mathbb{G}_T$
Pair	A pairing operation time
M_T	A multiplication operation time in $\mathbb{G}_T$
H	A hash operation time
m	The number of encrypting keywords
n	The number of attributes
t	The number of querying keywords
p	The order of groups $\mathbb{G}$ and $\mathbb{G}_T$
l	The number of data owners

TABLE 3: Computation cost comparison.

Algorithm	CP-ABKS [15]	ABKS-SM [19]	ABMKS-SM (ours)
KeyGen	$(2n + 2)E + nH$	$(2n + l + 4)E + E_T + H$	$(n + l + 3)E + H$
Encryption	$(2n + 4)E + nH$	$(2l + 2n + 2)E + 3E_T$	$(2l + n + 2)E + (m + 1)E_T + m\text{Pair} + mH + (m + 1)M_T$
Trapdoor	$(2n + 4)E$	$(2n + 1)E$	$E + tH$
Search	$(2n + 3)\text{Pair} + nE_T + (n + 2)M_T$	$(2N + 1)\text{Pair} + E_T + (2n + 1)M_T$	$3\text{Pair} + (t + 2)M_T$

TABLE 4: Storage cost comparison.

Algorithm	CP-ABKS [15]	ABKS-SM [19]	ABMKS-SM (ours)
KeyGen	$2n \log p$	$(2l + 2n)\log p$	$(2l + n)\log p$
Encryption	$2n \log p$	$(l + 2n)\log p$	$(n + l + m)\log p$
Trapdoor	$2n \log p$	$2n \log p$	$n \log p$

the computation overheads of key generation and encryption are on the increase as the number of attributes.

Figure 3 enlarges the time cost of the search algorithm in Figure 2. As illustrated in Figure 3, the time cost of the search process has almost unchanged as the number of attributes increases when the number of keywords is 10.

As illustrated in Figure 4, we can see that as the number of keywords grows larger, the time costs of key generation, encryption, trapdoor generation, search, and decryption process change when the number of the data user's attributes is 10. In this case, it is shown that the computation costs to generate keys, search, and decryption are minimal, while the costs of the encryption and trapdoor generation are on increase as the number of keywords.

Figure 5 enlarges the time cost of the search algorithm in Figure 4. As illustrated in Figure 5, when the number of attributes is 10, the time cost of the search process has almost unchanged as the number of keywords increases. This is because the number of keywords is not large enough and the multiplication operation M_T in group $\mathbb{G}_T$ is the lowest when compared to other operations.

In order to clearly compare the relationship between search time and keywords, we consider removing three pairings in search time due to the high cost of pairing. As illustrated in Figure 6, when the number of keywords becomes greatly larger, the computation cost of the search algorithm is proportional to the number of keywords when the attribute list satisfies the AND-gate access structure.

FIGURE 2: Performance evaluation for 10 keywords.

FIGURE 3: Search time for 10 keywords.

Furthermore, we show the time cost of decryption algorithm change as the number of data owners. As illustrated in Figure 7, when the number of data owners becomes larger, the computation overhead of the decryption algorithm is proportional to the number of data owners.

From the above figures, we can show that our search process is more efficient than other phases, and its

computation costs are free from the number of attributes. Although the computation cost of the search algorithm is proportional to the number of keywords when the attribute list satisfies the AND-gate access structure, the time cost of multiplication operation M_T in group $\mathbb{G}_T$ is the lowest. At the same time, the time cost for trapdoor generation has no relation with the number of attributes. Our scheme can

FIGURE 4: Performance evaluation for 10 attributes.

FIGURE 5: Search time for 10 attributes.

FIGURE 6: Search algorithm performance (except three pairing operations).

FIGURE 7: Decryption algorithm performance.

significantly enhance user's search experience, which corroborates benefits for data users to take advantage of Internet of things.

7. Conclusion

In this paper, we design a novel attribute-based multikeyword search for shared multiowner (ABMKS-SM) scheme in Internet of things and it can support to search multikeyword on ciphertexts with enhancing fine-grained access control. The most important security goal of public key searchable encryption is IND-CKA secure. We give this formal model definition and achieve IND-CKA security. Finally, we evaluate our performance with respect to functional, theoretical, and experimental analysis and further show our efficiency and practicality. Results demonstrate that time costs to search are free from the number of attributes; meanwhile, the costs for trapdoor generation are not related to the number of attributes. Last but not least, our proposed scheme makes full use of the benefits brought by cloud computing and Internet of things and is acceptable in practice.

Data Availability

The simulation result files used to support the findings of this study are available from the first author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Shandong Provincial Key Research and Development Program of China (no. 2018CXGC0701), the National Key Research and Development Program of China (no. 2018YFE0126000), the National Natural Science Foundation of China (NSFC) (no. 61972050), the Beijing Natural Science Foundation (no. L191012), and the 111 Project (no. B21049).

References

- [1] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 14, pp. 3690–3700, 2018.
- [2] H.-N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for internet of things: a survey," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8076–8094, 2019.
- [3] S. Xiong, Q. Ni, L. Wang, and Q. Wang, "Sem-acsit: secure and efficient multiauthority access control for iot cloud storage," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2914–2927, 2020.
- [4] S. Long, W. Long, Z. Li, K. Li, Y. Xia, and Z. Tang, "A game-based approach for cost-aware task assignment with QoS constraint in collaborative edge and cloud environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, 2020.
- [5] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding of the 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pp. 44–55, Berkeley, CL, USA, May 2000.
- [6] Y. Lu, J. Li, and Y. Zhang, "Privacy-preserving and pairing-free multirecipient certificateless encryption with keyword search for cloud-assisted IIoT," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2553–2562, 2020.
- [7] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceeding of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, Aarhus, Denmark, May 2005.
- [8] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 89–98, Alexandria, VI, USA, May 2006.
- [9] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07)*, pp. 321–334, Los Angeles, CL, USA, June 2007.
- [10] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.

- [11] J. Li, Y. Zhang, J. Ning et al., "vzAttribute based encryption with privacy protection and accountability for CloudIoT," *IEEE Transactions on Cloud Computing*, vol. 63, 2020.
- [12] J. Ning, X. Dong, Z. Cao, L. Wei, and X. Lin, "White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1274–1288, 2015.
- [13] Q. M. Malluhi, A. Shikfa, and V. C. Trinh, "A ciphertext-policy attribute-based encryption scheme with optimized ciphertext size and fast decryption," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pp. 230–240, Abu Dhabi, UAE, April 2017.
- [14] J. Li, W. Yao, Y. Zhang, H. Qian, and J. Han, "Flexible and fine-grained attribute-based data storage in cloud computing," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 785–796, 2016.
- [15] Q. Zheng, S. Xu, and G. Ateniese, "Vabks: verifiable attribute-based keyword search over outsourced encrypted data," in *Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pp. 522–530, Toronto, Canada, May 2014.
- [16] Q. Dong, Z. Guan, and Z. Chen, "Attribute-based keyword search efficiency enhancement via an online/offline approach," in *Proceedings of the 2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 298–305, Melbourne, Australia, December 2015.
- [17] J. Li, X. Lin, Y. Zhang, and J. Han, "Outsourced attribute-based encryption with keyword search function for cloud storage," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 715–725, 2016.
- [18] S. Qiu, J. Liu, Y. Shi, and R. Zhang, "Hidden policy ciphertext-policy attribute-based encryption with keyword search against keyword guessing attack," *Science China Information Sciences*, vol. 60, no. 5, Article ID 052105, 2017.
- [19] Y. Miao, X. Liu, K.-K. R. Choo et al., "Privacy-preserving attribute-based keyword search in shared multi-owner setting," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, 2019.
- [20] Y. Zhou, S. Zheng, and L. Wang, "Privacy-preserving and efficient public key encryption with keyword search based on cp-abe in cloud," *Cryptography*, vol. 4, no. 4, p. 28, 2020.
- [21] 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 Transactions on Computers*, vol. 65, no. 5, pp. 1566–1577, 2015.
- [22] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 4, pp. 1187–1198, 2016.
- [23] Z. Liu, Y. Liu, J. Xu, and B. Wang, "Privacy-preserving attribute-based multi-keyword search encryption scheme with user tracing," in *Proceedings of the International Symposium on Cyberspace Safety and Security*, pp. 382–397, Guangzhou, China, December 2019.
- [24] Y. Miao, J. Ma, X. Liu, F. Wei, Z. Liu, and X. A. Wang, "m2-abks: attribute-based multi-keyword search over encrypted personal health records in multi-owner setting," *Journal of Medical Systems*, vol. 40, no. 11, p. 246, 2016.
- [25] Y. Miao, J. Ma, X. Liu et al., "Vcksm: verifiable conjunctive keyword search over mobile e-health cloud in shared multi-owner settings," *Pervasive and Mobile Computing*, vol. 40, pp. 205–219, 2017.
- [26] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 506–522, Interlaken, Switzerland, May 2004.
- [27] Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," in *Proceedings of the International Conference on Pairing-Based Cryptography*, pp. 2–22, Tokyo, Japan, July 2007.
- [28] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proceedings of the I Theory of Cryptography Conference*, pp. 535–554, Amsterdam, the Netherlands, February 2007.
- [29] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Improved searchable public key encryption with designated tester," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pp. 376–379, Alexandria, VA, USA, April 2009.
- [30] J. Lai, X. Zhou, R. H. Deng, Y. Li, and K. Chen, "Expressive search on encrypted data," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, pp. 243–252, Hangzhou, China, May 2013.
- [31] Z. Lv, C. Hong, M. Zhang, and D. Feng, "Expressive and secure searchable encryption in the public key setting," in *Proceedings of the International Conference on Information Security*, pp. 364–376, New York City, NY, USA, December 2014.
- [32] R. Chen, Y. Mu, G. Yang, F. Guo, and X. Wang, "Dual-server public-key encryption with keyword search for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 789–798, 2015.
- [33] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," *Information Sciences*, vol. 403–404, pp. 1–14, 2017.
- [34] Y. Zhou, N. Li, Y. Tian, D. An, and L. Wang, "Public key encryption with keyword search in cloud: a survey," *Entropy*, vol. 22, no. 4, p. 421, 2020.
- [35] Y. Lu, J. Li, and Y. Zhang, "Secure channel free certificate-based searchable encryption withstanding outside and inside keyword guessing attacks," *IEEE Transactions on Services Computing*, vol. 24, 2019.
- [36] P. Xu, H. Jin, Q. Wu, and W. Wang, "Public-key encryption with fuzzy keyword search: a provably secure scheme under keyword guessing attack," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2266–2277, 2012.
- [37] W. Bing, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword search fuzzy search over encrypted data in the cloud," in *Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pp. 2112–2120, Toronto, Canada, May 2014.
- [38] H. Zhu, Z. Mei, B. Wu, H. Li, and Z. Cui, "Fuzzy keyword search and access control over ciphertexts in cloud computing," in *Proceedings of the Australasian Conference on Information Security and Privacy*, pp. 248–265, Perth, Australia, April 2017.
- [39] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in *Proceedings of the International Workshop on Information Security Applications*, pp. 73–86, Jeju Island, Korea, August 2004.
- [40] Y. Zhang and S. Lu, "Poster: efficient method for disjunctive and conjunctive keyword search over encrypted data," in *Proceedings of the 2014 ACM SIGSAC Conference on*

- Computer and Communications Security*, pp. 1535–1537, Scottsdale, AZ, USA, November 2014.
- [41] W. Sun, X. Liu, W. Lou, Y. T. Hou, and H. Li, “Catch you if you lie to me: efficient verifiable conjunctive keyword search over large dynamic encrypted cloud data,” in *Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 2110–2118, Hong Kong, China, April 2015.
 - [42] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, “Privacy-preserving multi-keyword ranked search over encrypted cloud data,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2013.
 - [43] Z. Xia, X. Wang, X. Sun, and Q. Wang, “A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2015.
 - [44] Y. Chen, W. Li, F. Gao, Q. Wen, H. Zhang, and H. Wang, “Practical attribute-based multi-keyword ranked search scheme in cloud computing,” *IEEE Transactions on Services Computing*, vol. 42, 2019.
 - [45] H. Wang, X. Dong, and Z. Cao, “Multi-value-independent ciphertext-policy attribute based encryption with fast keyword search,” *IEEE Transactions on Services Computing*, vol. 32, 2017.
 - [46] A. Wu, D. Zheng, Y. Zhang, and M. Yang, “Hidden policy attribute-based data sharing with direct revocation and keyword search in cloud computing,” *Sensors*, vol. 18, no. 7, p. 2158, 2018.
 - [47] B. Waters, “Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization,” in *Proceedings of the International Workshop on Public Key Cryptography*, pp. 53–70, Amsterdam, Netherlands, April 2011.
 - [48] A. Beimel, *Secure Schemes for Secret Sharing and Key Distribution*, Technion-Israel Institute of technology, Faculty of computer science, Haifa, Israel, 1996.
 - [49] A. De Caro and V. Iovino, “jpsc: Java pairing based cryptography,” in *Proceedings of the 2011 IEEE Symposium on Computers and Communications (ISCC)*, pp. 850–855, Corfu, Greece, July 2011.