ABSTRACT With the rapid growth of the cloud computing and strengthening of security requirements, encrypted cloud services are of importance and benefit. For the huge ciphertext data stored in the cloud, many secure searchable methods based on cryptography with keywords are introduced. In all the methods, attribute-based searchable encryption is considered as the truthful and efficient method since it supports the flexible access policy. However, the attribute-based system suffers from two defects when applied in the cloud storage. One of them is that the huge data in the cloud makes the users process all the relevant files related to the certain keyword. For the other side, the users and users’ attributes inevitably change frequently. Therefore, attribute revocation is also an important problem in the system. To overcome these drawbacks, an attribute-based ranked searchable encryption scheme with revocation is proposed. We rank the ciphertext documents according to the $\text{TF} \times \text{IDF}$ principle, and then only return the relevant top-$k$ files. Besides the decryption sever, an encryption sever is also introduced. And a large number of computations are outsourced to the encryption server and decryption server, which reduces the computing overhead of the client. In addition, the proposed scheme uses a real-time revocation method to achieve attribute revocation and delegates most of the update tasks to the cloud, which also reduces the calculation overhead of the user side. The performance evaluations show the scheme is feasible and more efficient than the available ones.

INDEX TERMS Attribute-based searchable encryption, attribute revocation, flexible access policy, rank.

I. INTRODUCTION With the development of social network and communication technology [1], data is showing explosive growth, which is a great challenge for devices with limited storage and computing resources. Therefore cloud service technologies [3] is rising rapidly. Increasingly, businesses and users are outsourcing their data to the cloud, greatly reducing local overhead. However, at the same time, data owners lose direct management of their data. And the data stored in the cloud is inevitable suffering from security threats. Therefore, for protecting the privacy and integrity of the data, the data owner must encrypt the data before outsourcing to prevent the leakage of private information to illegal users and the cloud.

But encrypted data will also greatly limit the availability of data. For example, how do data owners share their data with other legitimate users? The traditional encryption scheme can only realize one-to-one communication mode, and there is no doubt that the coarse-grained access mode cannot meet the development needs of today’s society. The attribute-based encryption proposed by Sahai and Waters, while protecting the confidentiality of data, implements a one-to-many access control mode, which is regarded as a very promising cryptography technology. Besides, how data users retrieve specific interested data files from encrypted data becomes a problem to be considered. The easiest way is to download all the ciphertext from the cloud, decrypt them into plaintext, and then find the required data. Obviously this requires a lot of storage and computing costs, which cannot meet the needs of users and the convenience of cloud computing.
Searchable encryption is a very effective method. The data user can generate query trapdoor based on the keyword of interest and match with ciphertext index without decryption. If the match is successful, all encrypted files containing the keyword will be returned to the user. However, not all the returned results match the user’s request. In a healthcare system, for example, the attending physician may want to know more about the patient’s physical health than mental or other conditions. Therefore, it is necessary to design a ranked searchable encryption. The cloud ranks the query results according to the relevant scores of the query keyword and the file. Only the $k$ files closest to the user’s request are returned, which not only saves the consumption of network bandwidth, but also reduces the time cost of users, in line with today’s accelerating life style. In addition, the user’s attribute may change frequently, so attribute revocation must be considered.

A. CONTRIBUTIONS OF THE SCHEME

To address the above problems, we present a scheme which supports ranked keyword search and attribute revocation in this paper. The highlights of the programme are shown below:

- **Efficient search:** Since the traditional single-keyword search may return a large number of redundant files, the search accuracy cannot meet the needs of users. We adopt the TF-IDF principle to rank the search results and only return the relevant top-$k$ files, greatly improving the accuracy of the search.

- **Low computational overhead on the client side:** To solve the problem of the high computing cost for the client, we introduce the Encryption-Server Providers (E-SP) and the Decryption Server Providers (D-SP), and outsource a large number of computational tasks in the encryption and decryption phases, respectively, which lightens the computing burden of the client.

- **Attribute revocation:** For the dynamic characteristics of the system, we use a real-time attribute revocation method. The private key component related to the attribute is outsourced to D-SP, and the user only stores the private key component that is not related to the attribute, which reduces the storage overhead of the user side. Besides, when the user’s attribute is revoked, a large number of update calculations will be transferred to the cloud, reducing the computational overhead on the user side.

B. ORGANIZATION

The remaining of the paper is shown as follows. We first introduce some related works about ranked keyword search, attribute-based encryption and attribute revocation in Section 2. We review some basic knowledge in Section 3, which include bilinear map, access structure, linear secret-sharing scheme, relevance score function and the complexity assumption. In Section 4, the symbols used in the scheme will be introduced first, followed by the definition of the scheme as well as the system and security model. In Section 5, we show the specific structure of the scheme. The security of the algorithm is proved in Section 6. The performance evaluation is shown in Section 7. And we give a brief summary at the end of this paper.

II. RELATED WORK

A. RANKED KEYWORD SEARCH

Searchable encryption (SE) is an important method of securely and efficiently searching for ciphertext in the cloud. Song et al. [4] first introduced a symmetric keyword search encryption algorithm in 2000 that implements the function of retrieving ciphertext. Dan et al. [5] constructed an asymmetric searchable encryption algorithm in 2004. Subsequently, Sun et al. [6] introduced an attribute-based keyword search scheme in which the size of trapdoors generated by legitimate users is constant, and the computational cost of the encryption stage is kept at a constant level. Later, Miao et al. [8] proposed a data sharing system supporting online/offline encryption in 2019, which supports the outsourcing of complex computing tasks in the decryption phase to cloud servers. But for keyword search, not all returned results are within the user’s needs. Hence, the query results should be ranked according to certain conditions, and only the top-$k$ files that best meet the user’s request are returned. In 2015, Revathy et al. [9] introduced an attribute-based searchable encryption based on KNN technology, which can rank the search results and return the top-k query results. Subsequently, Jiang et al. [10] constructed a ranked searchable algorithm based on TF×IDF rules to sort the query results, but the scheme does not support user fine-grained access control over encrypted data. Liu et al. [11] proposed a verifiable ranked searchable encryption scheme which supports search result verification while avoiding redundant results.

B. ATTRIBUTE-BASED ENCRYPTION

Sahai and Waters [17] first introduced an attribute-based encryption (ABE) system in 2005, which can support fine-grained access control over the ciphertext data. Sahai et al. started the beginning of ABE widely studied by researchers. Then Goyal and Bethencourt et al. proposed two different variant mechanisms based on ABE: key policy attribute-based encryption (KP-ABE) [19] and ciphertext policy attribute-based encryption (CP-ABE) [20]. In the KP-ABE scheme, the users’ private key corresponds to an access structure, and the ciphertext corresponds to an attribute set. The decryption will be successful if and only if the attribute set satisfies the access policy. The algorithm idea of CP-ABE is opposite to that of KP-ABE. Therefore, the KP-ABE scheme is applicable to query scenarios, like
Digital Rights Management System, the CP-ABE scheme is applicable to access control scenarios, like electronic medical systems. Since then, some scholars have extended the classic ABE scheme in different aspects according to actual needs, so that it can meet different application needs, such as searchable encryption, security outsourcing, attribute revocation in [30]–[35].

C. ATTRIBUTE REVOCATION

To ensure the practicability and security of cloud storage, attribute revocation is a problem that must be considered. Attribute revocation means that the user has lost the corresponding access control rights. In 2006, Pirretti et al. [21] first proposed an attribute revocation scheme for ABE. By setting the expiration date for the attribute, the authority periodically updates the attribute version and revokes the user attribute by revoking the latest version of the attribute. Subsequently, Ibraimi et al. [22] introduced a ciphertext policy attribute-based encryption scheme with instant attribute revocation. Wang et al. [23] constructed an attribute revocation scheme that assigns a random number, called the version number, to the ciphertext and the user’s secret key. When the revocation occurs, the trusted authority randomly assigns a new version number. Then the cloud and the user perform ciphertext and key update algorithms with the new version number. Hur and Noh [24] proposed a CP-ABE scheme with immediate attribute revocation, which distributes the KEK binary tree to users, but the scheme has a large maintenance overhead. In 2018, Liu et al. [25] proposed a large universe attribute-based encryption scheme that supports both policy update and attribute revocation. Wang et al. [26] introduced an efficient attribute revocation scheme, which greatly reduced the computational overhead of the client by entrusting the ciphertext and user secret key update tasks to the cloud. The problem of attribute revocation has been widely concerned and studied. The majority of scholars are committed to improving efficiency and security in the revocation process.

III. PRELIMINARIES

Some preliminaries will be introduced in this section, including bilinear map, access structures, linear secret sharing schemes(LSSS), relevance score function, generic bilinear group and complexity assumptions.

A. BILINEAR MAP

Given two cyclic groups $G_1$ and $G_T$ are given two cyclic groups which enjoy the same prime order $p$, $e : G_1 \times G_1 \rightarrow G_T$ is a bilinear map if it has the properties below.

1) Bilinearity: For any $\bar{h}_1, \bar{h}_2 \in G_1$ and any $\chi_1, \chi_2 \in Z_p$, $e(\bar{h}_1^{\chi_1}, \bar{h}_2^{\chi_2}) = (\bar{h}_1, \bar{h}_2)^{\chi_1 \chi_2}$.
2) Non-degeneracy: For $\bar{h}_1, \bar{h}_2 \in G_1$, $e(\bar{h}_1, \bar{h}_2) \neq 1$
3) Computability: For any $\bar{h}_1, \bar{h}_2 \in G_1$, there must exist an algorithm to calculate $e(\bar{h}_1, \bar{h}_2)$ efficiently.

B. ACCESS STRUCTURE

Given a group of $t$ participants, denoted as $\Omega = \{P_1, P_2, \ldots, P_t\}$. A collection $C \subseteq 2^\Omega$ is monotone if $X \subseteq Y$ and $X \subseteq \emptyset$ then $Y \subseteq \emptyset$ for all $X, Y$. An access structure is a collection $C$ of non-empty subsets of $\Omega = \{P_1, P_2, \ldots, P_t\}$. Simply speaking $C \subseteq 2^\Omega \setminus \emptyset$. The sets in $C$ are considered as an authorized sets, the sets not in $C$ are treated as unauthorized sets.

C. LINEAR SECRET SHARING SCHEMES(LSSS)

A linear secret-sharing scheme $\Pi$ defined upon the group $\Omega$ has two properties below:

1) The share of the secret $s$ for each participant forms a vector on $Z_p$;
2) There is a shared-matrix $A$ with $l$ rows and $n$ columns. Let $\rho$ be a function which maps from $\{1, \ldots, l\}$ to $\Omega$, namely $\rho$ maps every row of matrix $A$ to a participant. We choose the column vector $\bar{r} = (s_1, s_2, \ldots, s_n)$ randomly, where $s \in Z_p$ is the shared secret value, and $s_2, \ldots, s_n$ are selected in $Z_p$ randomly, then $A^\top \bar{r}$ denotes the vector of $l$ shares of the secret $s$ according to $\Pi$. Let $A_i$ represent the $i$th row of $A$, then $\lambda_i = A_i^\top \bar{r}$ denotes the share belonging to party $\rho(i)$.

Linear reconstruction: Each LSSS $\Pi$ enjoys the property of linear reconstruction. Assume that pairing $(A, \rho)$ represents an access structure $C$ for LSSS $\Pi$ and $S$ is an authorized set in $C$, i.e., $C$ contains $S$. There must exist constants $w_i \in Z_p$ such that $\sum_{i \in l} w_i \lambda_i = s$ for shares $\{\lambda_i\}$ of secret $s$, where $I = \{i : \rho(i) \in S\} \subseteq [1, l]$.

D. RELEVANCE SCORE FUNCTION

We consider using a widely used statistical metric to implement the ranking function. Generally, we think that the more frequently the query keyword arises in the file, the more relevant it is to the file. Actually not. Because the importance of distinct keywords varies in the file collection. Some very basic keywords can not convey the meaning of the file. Here we adopt the Term Frequency-Inverse Document Frequency technique to measure the relationship between keywords and files, where Term Frequency(TF) means the frequency of a query keyword arises in the file and the Inverse Document Frequency(IDF) can be computed by dividing the number of file collections by the number of files matching the certain keyword. When gaining the TF and IDF values, we multiply these two values to obtain the TF×IDF value of the specified keyword, called the relevance score. The larger the TF×IDF value of a certain keyword, the more relevant it is to the document.

In the proposed scheme, we choose a general calculation method for correlation scores and redefine TF and IDF. The IDF value of any keyword $w_i$ in file $F_{\ell}$ is computed as...
IDF_{τ,ϱ} = \ln(1 + \frac{d}{|F(τ)|}). The TF value of \( w_τ \) in file \( F_0 \) is computed as \( TF_{τ,ϱ} = (1 + \ln f_{τ,ϱ}) \), where \( f_{τ,ϱ} \) represents the number of the keyword \( w_τ \) in \( F_0 \), and \( |F_0| \) is a normalization factor gained by \( \sqrt{\sum_{τ=1}^n (1 + \ln f_{τ,ϱ})^2} \). Then, the relevance score between an interested keyword \( w_τ \) and the file \( F_0 \) is defined as

\[
TF \times IDF_{τ,ϱ} = \frac{(1 + \ln f_{τ,ϱ})}{|F_0|} \ln(1 + \frac{d}{|F(τ)|})
\]

E. GENERIC BILINEAR GROUP
Suppose \( ψ_1, ψ_2 \) are two random encodings of the group \( Z_p \). And \( ψ_1, ψ_2 \) are injective functions which map from \( Z_p \) to \( \{0, 1\}^θ \), with \( θ > 3\log(p) \). Let \( G_1 = \{ψ_1(η) | η ∈ Z_p\} \) and \( GT = \{ψ_2(η) | η ∈ Z_p\} \), where \( G_1 \) represents a generic bilinear group. Let \( g \) indicates \( ψ_1(1) \), and \( g^θ \) means \( ψ_1(η) \), \( e(g, g) \) indicates \( ψ_2(1) \) and \( e(g, g^θ) \) means \( ψ_2(θ) \).

F. COMPLEXITY ASSUMPTION
The definition of the decisional \( q \)-parallel bilinear Diffie-Hellman exponent problem (\( q \)-DBDHE) is given first below. Suppose \( g \) is a generator of group \( G_1 \). Given

\[
Φ = (g, g^θ, g^a, g^{a^2}, g^{a^3}, g^{a^4}, g^{a^5}, g^{a^6}, g^{a^7}, g^{a^8}, g^{a^9}, g^{a^{10}}, g^{a^{11}}, g^{a^{12}}, g^{a^{13}}, g^{a^{14}}, g^{a^{15}}, g^{a^{16}}, g^{a^{17}}, g^{a^{18}}, g^{a^{19}}, g^{a^{20}})
\]

where \( a, b_1, \ldots, b_2 ∈ Z_p \) are selected randomly and \( k, j \in \{1, \ldots, q\} \), \( k \neq j \).

There is no polynomial time algorithm \( A \) which can effectively distinguish tuples \( e(g, g)^{a^{k+j}} \) and element \( Z \) with a non-negligible advantage \( ϵ_1 \), where \( Z \) is selected in \( G_T \) randomly. In short, \( A \)’s advantage \( Adv_A \) in solving the \( q \)-DBDHE problem is negligible, which can be denoted as follows.

\[
|Pr[A(Φ, e(g, g)^{a^{k+j}}) = 0] - Pr[A(Φ, Z) = 0]| < ϵ_1
\]

Definition 1 (\( q \)-DBDHE Assumption): We say if the advantage of the probabilistic polynomial time algorithm in solving the \( q \)-DBDHE problem can be negligible, the \( q \)-DBDHE assumption holds.

IV. SYSTEM AND SECURITY MODEL
A. NOTATIONS

| Notations | Definitions |
|-----------|-------------|
| \( U \) | Attribute universe in the system |
| \( S \) | Attribute set of data user |
| \( F \) | File set of data owner |
| \( Δ \) | Keyword dictionary constructed by Data owner |
| \( CT' \) | Intermediate ciphertext |
| \( CT \) | Ciphertext file |
| \( E \) | Partial decryption ciphertext |
| \( I_Δ \) | Secure indexes of the dictionary |
| \( w \) | An interested keyword of data user |
| \( T D_w \) | Data user’s query trapdoor |
| \( RL_w \) | The attribute revocation list |

B. SYSTEM MODEL
As illustrated in Fig.1, there exists six types of entities in the proposed scheme: a trusted authority, a cloud server provider, an encryption-server provider, a decryption-server provider, data owners, data users.

- **Trusted Authority (TA):** The TA is fully trusted by other entities. And it is responsible for generating the system public parameters and master keys. Besides, the TA is responsible for revoking users’ attributes based on his dynamic roles.
- **Data Owner (DO):** The DO is responsible for local encryption and uploading the ciphertext data to the
Cloud. In the meantime, the DO uses the text processing system to obtain the key dictionary and generate the keyword index array, which are uploaded to the cloud together with ciphertext.

- **Cloud Server Provider (CSP):** The CSP stores encrypted data and performs data search. Once the CSP receives a query trapdoor for any specified keyword, it will test whether the query trapdoor and the keyword index match. The CSP is also responsible for a large number of calculation tasks in the revocation phase, such as ciphertext update tasks.

- **Encryption-Server Provider (E-SP):** The E-SP has powerful computing ability. It uses the system public keys and the access policy specified by the DO to perform initial encryption to generate the intermediate ciphertext, and returns it to the DO.

- **Decryption-Server Provider (D-SP):** Once receiving the corresponding top-\( k \) encrypted files from the CSP, the D-SP performs the partially decryption algorithm if and only if the attribute set satisfies the access policy. In addition, it is also responsible for updating users' delegated secret key in the revocation phase.

- **Data User (DU):** The DU performs local decryption with his private key component, and finally obtains the top-\( k \) files he/she is interested in.

### C. FORMAL DEFINITION

The formal definition of outsourcing attribute-based ranked searchable encryption with revocation is shown below.

**Setup**(\( 1^k, \mathcal{U} \)) \( \rightarrow (PK, MSK) \): Inputting a security parameter \( 1^k \) and an attribute universe \( \mathcal{U} \), TA generates the system public keys \( PK \) and the master keys \( MSK \).

**KeyGen**(\( MSK, S \)) \( \rightarrow SK \): TA executes the algorithm for gaining the user’s secret key. It inputs \( MSK \) and \( S \) to generate a private key \( SK = \{SK_k, SK_h\} \), where \( SK_k \) will be sent to D-SP, and \( SK_h \) will be sent to DU.

**Outsource encryption**(\( PK, (A, \rho) \)) \( \rightarrow CT' \): E-SP performs the outsourced encryption algorithm. Taking \( PK \) and \( (A, \rho) \) as input, the algorithm generates the intermediate ciphertext \( CT' \).

**File encryption**(\( PK, CT', F \)) \( \rightarrow CT \): The encryption algorithm is executed by DO. Inputting \( PK \), the intermediate ciphertext \( CT' \) and the file set \( F = \{F_1, F_2, \ldots, F_d\} \), DO outputs the ciphertext \( CT \), where \( d \) represents the number of files.

**Index generation**(\( PK, \Delta \)) \( \rightarrow I_\Delta \): The algorithm is performed by DO. Inputting \( PK \) and the keyword dictionary \( \Delta \), DO outputs the indexes \( I_\Delta \) and sends it to CSP along with the ciphertext.

**Trapdoor generation**\( w, SK_u \) \( \rightarrow TD_w \): The algorithm is performed by DU. Inputting an interested keyword \( w \) and the private key component \( SK_u \), DU obtains a query trapdoor \( TD_w \).

**Search**\( I_\Delta, TD_w \) \( \rightarrow 1 \) or \( \perp \): Inputting the index \( I_\Delta \) and the query trapdoor \( TD_w \), the server calls the search algorithm.

If it is matched, this algorithm outputs 1 and sends the corresponding index array to D-SP, otherwise \( \perp \).

**Outsource decryption**\( (CT_{ABE}, SK_u) \) \( \rightarrow E \): The outsourced decryption algorithm is performed by D-SP. Inputting the public parameters \( PK \), private key component \( SK_u \) associated with an attribute set \( S \) and \( CT_{ABE} \) related to the access policy \( (A, \rho) \), it generates the partially decrypted ciphertext \( E \) if and only if \( S \) satisfies \( (A, \rho) \).

**Decryption**\( (E, SK_u) \) \( \rightarrow F_\theta \): Inputting the partially decrypted ciphertext \( E \) and the private key component \( SK_u \), DO performs local decryption to get the file \( F_\theta \) he/she is interested in.

**UkeyGen**\( (PK, MSK, \theta') \) \( \rightarrow (AVK_{\theta'}, APK_{\theta'}) \): The algorithm is executed by TA. It takes \( PK \), \( MSK \) and the canceled attribute \( \theta' \) as input and outputs updated attribute version key \( AVK_{\theta'} \) and updated attribute public key \( APK_{\theta'} \). Then TA sends \( AVK_{\theta'} \) and \( RL_{\theta'} \) to D-SP and CSP to perform updates of secret key and ciphertext.

**SKUpdate**\( (SK_u, AVK_{\theta'}) \) \( \rightarrow (SK_{u}') \): D-SP performs the algorithm to update the delegate secret key component \( SK_u \). It takes \( SK_u \) and updated attribute version key \( AVK_{\theta'} \) as input and generates an updated secret key component \( SK_{u}' \).

**CTUpdate**\( (CT_{ABE}, AVK_{\theta'}) \) \( \rightarrow CT_{ABE}' \): CSP executes the ciphertext update algorithm. Inputting \( CT_{ABE} \) related to the canceled attribute \( \theta' \) and updated attribute version key \( AVK_{\theta'} \), it generates an updated ciphertext \( CT_{ABE}' \).

### D. SECURITY MODEL

Assumed that in the system, the CSP, E-SP and D-SP are honest but curious. To be brief, they will strictly follow the user’s instructions, but they will do everything possible to get more information. So we consider two types of adversaries in the proposed scheme.

- **Type-I adversary** is defined as the curious user colluding with D-SP and CSP. They are allowed to interrogate the private key components \( SK_k \) and the private keys \( SK_h \) of dishonest users. The main purpose of such adversaries is to obtain some interested information from ciphertext that do not belong to them.

- **Type-II adversary** is defined as the curious user colluding with E-SP, which can obtain encryption parameters information and ciphertext for malicious users.

The proposed outsourcing attribute-based ranked searchable encryption with attribute revocation is indistinguishability against selective ciphertext-policy and chosen plaintext attack(IND-sCP-CPA) secure and indistinguishability against selective chosen keyword attack (IND-CKA) secure. To describe the security, we define the following interactive games between the challenger \( B \) and an adversary \( A \).

The security game for the type-I adversary is denoted below.

**Init.** The adversary \( A \) announces a challenging access policy \( A^* \), and passes it to the challenger \( B \).

**Setup.** \( B \) first runs the **Setup** algorithm to outputs the public keys \( PK \) and the master keys \( MSK \), then transmits \( PK \) to \( A \).
Phase I. $A$ can make queries to the following oracles:

- $O_{SK}(S)$: $A$ can make any private key queries about attribute $S$ which can not satisfy the specified access policy $A'$.  
- $O_{UK}(\theta)$: $A$ can make any queries for updated key related to the canceled attribute $\theta$.

Challenge. $A$ chooses two messages $R_1, R_2$ randomly which are equal length. $B$ picks $r \in \{0, 1\}$ randomly, then performs encryption algorithm for $R_r$ under $A'$ to obtain the challenge ciphertext $CT^*$. Finally, $B$ return $CT^*$ to $A$.

Phase II. $A$ makes queries similar to phase I, $B$ answers like phase I.

Guess. $A$ returns a guess $r'$ and wins this game if $r' = r$. The $A'$s advantage of breaking the game is set as

$$Adv_A = |Pr[r' = r] - 1/2|$$

The game for the type-II adversary is similar to the above game.

Definition 2: If the advantage of winning the above game for any polynomial time adversary is negligible, then outsourcing attributed-based ranked searchable encryption with revocation scheme is IND-sCP-CPA secure.

For the security of the keyword index, we define the indistinguishability against the selective chosen keyword attack (IND-CKA) game below.

Setup. $B$ performs the Setup algorithm to gain $PK$, then transmits it to $A$.

Phase I. $A$ can inquire the $O_{SK_0}$ and $O_{TD_0}$ for the secret key and the trapdoor in polynomial time.
- $O_{SK_0}(PK, MSK)$: Inputting $PK$ and $MSK$, $B$ generates the secret key $SK_0$ and transmits it to $A$.
- $O_{TD_0}(PK, w^*, SK_0)$: $A$ submits an interested keyword $w^*$ for trapdoor query. $B$ uses $PK$, $SK_0$ and $w^*$ to generate trapdoor and sends it to $A$.

Challenge. $A$ picks two equal-size keywords $w_0, w_1$ with the restriction that $w_0$ and $w_1$ were not previously used for trapdoor queries. $B$ picks $r \in \{0, 1\}$, then gets the challenge index $h_r$, and sends it to $A$.

Phase II. $A$ makes queries similar to phase I, but the only limitation is $w \neq w_0, w_1$, and $B$ answers like phase I.

Guess. $A$ gives a guess $r'$ and wins this game if $r' = r$. The $A'$s advantage of winning the above game is set as

$$adv_A = |Pr[r' = r] - 1/2|$$

Definition 3: If the advantage of winning the above game for any polynomial time adversary is negligible, then outsourcing attributed-based ranked searchable encryption with revocation scheme is IND-CKA secure.

V. CONCRETE CONSTRUCTION

Aiming to reduce the computational overhead of the clients, the encryption server and the decryption server were introduced in the system. In addition, to solve the resource consumption caused by single-keyword search, we rank the search results according to the TF×IDF principle and only return the top-$k$ files. The concrete structure of the proposed scheme is shown as follows:

A. SYSTEM INITIALIZATION

Setup $(1^\lambda, \mathcal{U}) \rightarrow (PK, MSK)$: Inputting a security parameter $\lambda$ and the universe set of attribute $\mathcal{U} = \{\theta_1, \ldots, \theta_n\}$, TA chooses two cyclic groups $G_1$ and $G_T$ with the same prime order $p$. Then it defines a bilinear map $e: G_1 \times G_1 \rightarrow G_T$ and four collision resistant hash functions:

$$H : \{0, 1\}^* \rightarrow Z_p^* \quad H_0 : \{\mathcal{M}\} \rightarrow \{0, 1\}^{nH_0}$$

where $\mathcal{M}$ denotes the plaintext space, $nH_0$ and $nH_2$ are the output-sizes of hash functions $H_0$ and $H_2$, respectively. TA randomly selects numbers $a, x, \tilde{x} \in Z_p$ and computes $Y = e(g, g)^x, B_0 = g^a, B_1 = g^{\tilde{x}}$ where $g$ is the generator of $G_1$. Then it randomly selects $v_\theta \in Z_p$ for each $\theta \in \mathcal{U}$ as the attribute version key, that is $AVK_\theta = v_\theta$. And it computes the attribute public key $APK_\theta = g^{v_\theta}$. Finally the algorithm generates the following system public keys $PK$ and master keys $MSK$.

$$PK = \{G_1, G_T, p, e, g, H, H_0, H_1, H_2, Y, B_0, B_1, \{APK_\theta\}_{\theta \in \mathcal{U}}\}$$

$$MSK = \{x, \tilde{x}, \{v_\theta\}_{\theta \in \mathcal{U}}\}$$

B. KEY GENERATION

KeyGen $(MSK, S) \rightarrow SK$: Taking $MSK$ and an attribute set $S = \{\theta_1, \ldots, \theta_s\}$ as input, the algorithm generates the private key $SK$. Firstly, TA chooses two numbers $x_1, x_2 \in Z_p$ randomly such that $x = x_1 + x_2 \mod p$. Then it selects $t, u \in Z_p$ and sets the secret key as $SK = \{SK_x, SK_u\}$, where

$$SK_x = \{K_{x_1} = g^{x_1/u}, K_{x_2} = g^{x_2/u}, K_\theta = \{g^{[H(\rho)^x/u]}_{\theta \in S}\}$$
$$SK_u = \{K_{u_1} = g^{u}, K_{u_2} = u, K' = g^{\tilde{x}}\}$$

Then $SK_x$ will be sent to D-SP and $SK_u$ to the user.

C. ENCRYPTION

Outsource Encryption $(PK, (A, \rho)) \rightarrow CT'$: The algorithm inputs $PK$ and the access policy $(A, \rho)$, where $\rho$ is a function which maps rows of the matrix $A$ to attributes. It selects $r_i \in Z_p$ randomly for each $i \in \{1, \ldots, l\}$ and sets

$$C_i' = g^{\rho H(\rho(x))}, D_i' = g^{\rho(y)}$$

Finally the algorithm generates the intermediate ciphertext $CT' = \{C_i', D_i'\}_{i \in \{1, \ldots, l\}}$.

File Encryption $(PK, CT', F) \rightarrow CT$: The algorithm consists of the following two steps:

$(A_i)$ DO inputs $PK, CT'$ and a message $R$, where $R$ is selected randomly from plaintext space $M$. And DO chooses a vector $\tilde{r} = \{s, x_2, \ldots, s_0\} \in Z_p$ where $s$ is the shared secret value. $A_i$ stands for the $i$th row of $A$ and it calculates $\lambda_i = A_i \cdot \tilde{r}$ for $\forall i \in \{1, \ldots, l\}$. The algorithm calculates:

$$CT_{ABE} = \{C_0, C_1, \{C_i, D_i\}_{i \in \{1, \ldots, l\}}\}$$
D. INDEX GENERATION

IndexGen (PK, \Delta) \rightarrow I_\Delta: Do sends the file set \mathcal{F} = \{F_1, F_2, \ldots, F_d\} to the text processing system (TPS). Then TPS scans and analyzes the files to generate a keyword dictionary \Delta, and calculates the relevant scores of keywords and files. Finally, the relevant score table (RST) is returned to the user in the following form.

Then DO builds secure indexes table by following these steps:

1. **Setup:**
   1. For any keyword \( w_\tau \in \Delta \), build a list \( \mathcal{F}(w_\tau) \), and set \( \mathcal{F}(w_\tau) = \{id(F_{\tau_1}), id(F_{\tau_2}), \ldots, id(F_{\tau_o})\} \) with \( |\mathcal{F}(w_\tau)| = o; \)
   2. For every file identifier \( id(F_\theta) \in \mathcal{F}(w_\tau) \), compute the relevance score of the file \( F_\theta \) to the keyword \( w_\tau \).

2. **Construct the query arrays:**

As shown in Fig.3, for every keyword \( w_\tau \in \Delta \), let \( max \) be the maximum size of each array \( A_i \).

- For \( q = 1, \ldots, o \), construct a node
  \[
  N_{\tau,q} = (\langle id(F_{\tau_q})\|TF - IDF_{\tau,q}\|\xi >, r_{\tau,q})
  \]

where \( id(F_{\tau_q}) \) is the \( q \)th file identifier in \( \mathcal{F}(w_\tau) \). \( TF - IDF_{\tau,q} \) is the relevant score of the keyword \( w_\tau \) and the file \( F_{\tau_q} \). \( r_{\tau,q} \) denotes a \( \lambda \)-length random string. \( \xi \in \{0, 1\} \) represents whether \( id(F_{\tau_q}) \) is the last file in \( \mathcal{F}(w_\tau) \).

L. Zhang et al.: Outsourcing Attributed-Based Ranked SE With Revocation for Cloud Storage

FIGURE 2. Index array model.

FIGURE 3. Relevant scores table.

Next DO encrypts the arrays through a symmetric encryption algorithm such as AES.

3. **Generate index:**

The algorithm inputs the system public parameters \( PK \) and the dictionary \( \Delta = \{w_1, w_2, \ldots, w_m\} \). For every keyword \( w_\tau \in \Delta (\tau = 1, \ldots, m) \), DO randomly chooses \( \xi_\tau \in Z_p \), and calculates \( l_1 = g^{\xi_\tau} \). Then he/she chooses \( \pi \) and computes

\[
I_\tau = g^{\xi_\tau} T_1^{\pi}, T_2 = g^{\alpha \pi}, T_3 = g^{\xi_\tau} T_3^{\pi}. \]

The algorithm outputs the following secure keyword index for the dictionary \( \Delta \).

\[
I_\Delta = \{I_{w_\tau} = (I_1, I_2, I_3)\}_{1 \leq \tau \leq m}
\]

E. TRAPDOOR GENERATION

TrapGen (w, SK_w) \rightarrow TD_w: DU inputs SK_w and a query keyword \( w \) which he/she is interested in to generate a query trapdoor \( TD_w \). DU selects \( \alpha \in Z_p \) randomly and calculates:

\[
T_1 = g^{\alpha \tau}, T_2 = g^{\alpha \tau}, T_3 = g^{\alpha \tau}. \]

The algorithm outputs the trapdoor as \( TD_w = \{T_1, T_2, T_3\} \).

F. SEARCH

Search (I_\Delta, TD_w) \rightarrow 1 or 0: CSP locates the array matching the trapdoor \( TD_w \). It checks whether the formula below holds for any keyword index \( I_{w_\tau} \in I_\Delta \).

\[
e(I_1, T_1) e(I_2, T_3) = e(I_2, T_2)\]

If the above formula holds, it outputs 1 and sends the array to D-SP. D-SP decrypts the array using the AES algorithm, retrieves the top-k matching documents based on the TF-IDF rule and then sends the file identifier to CSP. Finally CSP sends the search results to D-SP for partial decryption.

G. DECRYPTION

Outsource Decryption (CT, SK_s) \rightarrow \mathcal{E}: The algorithm inputs the system public parameters \( PK \), private key component \( SK_s \) associated with the attribute set \( s \) and \( CT_{ABE} \) related to access policy \( \mathcal{A}, \rho \). When \( s \) can not satisfies \( \langle A, \rho \rangle \), it outputs \( \bot \).
and terminates request. Otherwise D-SP performs the algorithm as shown below.

The algorithm defines \( I \subset \{1, 2, \ldots, l\} \) as \( I = \{i : \rho(i) \in S\} \), and then selects a tuple of constants \( \{\tilde{w}_i \in Z_p\}_{i \in I} \) such that \( \sum_{i \in I} \tilde{w}_i \lambda_i = s \) for shares \( \{\lambda_i\} \) of any secrets \( s \). It computes:

\[
E = \frac{\prod_{i \in I} e(C_i, K_2)}{e(C_1, K_1) \prod_{i \in I} e(D_i, K_{\rho(i)})} = e(g, g)^{\gamma s / u}.
\]

Finally, D-SP obtains the partially decrypted ciphertext \( E \) and returns it to DU.

**Decryption** \( (E, SK_u) \rightarrow F_\theta \). Once receiving \( E \) from D-SP, DU inputs the secret key component \( SK_u \). And the random message \( R \) is computed as:

\[
R = \frac{C_0 \cdot E^{K_{u,2}}}{e(C_1, K)}
\]

Then DU computes \( R_0 = H_0(R) \) and checks whether \( H_2(R_0)|CT_{sym}(F_\theta)| = VK_\theta \) is established. If the equation holds, DU computes \( S_{sym} = H_1(R_0) \). Finally, DU obtains the top-\( k \) files \( F_\theta = DEC_{sym}(S_{sym}, CT_{sym}) \). Otherwise the algorithm outputs \( \bot \).

**H. ATTRIBUTE REVOKEATION**

If any users’ attribute \( \theta' \) is revoked in the system, TA first puts those identities whose attribute is canceled into \( RL_{\theta'} \). Then and after it executes the following algorithm.

1) **UkeyGen**(PK, MSK, \( \theta' \)) \rightarrow (AVK_{\theta'}, APK_{\theta'}): Inputting PK, MSK as well as the canceled attribute \( \theta' \), TA chooses a new value \( \bar{v}_{\theta'} \in Z_p (\bar{v}_{\theta'} \neq v_{\theta'}) \) randomly for the canceled attribute \( \theta' \), and calculates the updated attribute version key as

\[
AVK_{\theta'} = \bar{v}_{\theta'}/v_{\theta'}.
\]

In addition, TA computes the updated attribute public key as

\[
APK_{\theta'} = (APK_{\theta'})^{AVK_{\theta'}} = (APK_{\theta'})^{\gamma s / \gamma'} = g^{\gamma s / \gamma'}.
\]

Finally, TA declares the updated attribute public key \( APK_{\theta'} \). Then, TA passes the updated attribute version key \( AVK_{\theta'} \) and the attribute revocation list \( RL_{\theta'} \) to D-SP and CSP to perform the update tasks of delegated key components and ciphertext.

2) **SkeyUpdate**(SK, AVK_{\theta'}) \rightarrow (SK'_{\theta}) : Inputting the the delegate secret key component \( SK \) and the updated attribute version key \( AVK_{\theta'} \), D-SP updates the new key \( SK'_{\theta} \) as

\[
K_{s,1} = K_{s,1} = g^{\gamma s / \gamma'}, \quad K_{\rho,2} = K_{\rho,2} = g^{\gamma / \gamma'}, \quad K_{\theta} = \left\{ \begin{array}{ll} (g^{\gamma s / \gamma'})^{AVK_{\theta}} & , \theta \neq \theta' \\ (g^{\gamma / \gamma'})^{AVK_{\theta}} & , \theta = \theta' \end{array} \right.
\]

3) **CTUpdate**(CT_{ABE}, AVK_{\theta'}) \rightarrow CT_{ABE} : Inputting CT_{ABE} associated with the canceled attribute \( \theta' \) and AVK_{\theta'}, CSP updates CT_{ABE} as

\[
\widetilde{C}_0 = C_0 = R \cdot e(g, g)^x, \quad \widetilde{C}_1 = C_1 = g^x, \quad \widetilde{C}_i = C_i = g^{\gamma s / \gamma'} \cdot e(g, g)^{\gamma s / \gamma'}, \quad \widetilde{D}_i = D_i = g^{\gamma s / \gamma'}, \quad \rho(i) = \theta' \}
\]

It’s worth noting that:

1) Only the attribute-related delegate secret key components \( SK_{\rho} \) need to be updated, and the user’s private key components \( SK_u \) remain unchanged, greatly reducing the burden on the client.

2) CSP stores the updated attribute version key \( AVK_{\theta'} \). When the user accesses the ciphertext, CSP first checks whether any attribute embedded in the ciphertext is revoked. If attribute revocation occurs, CSP performs ciphertext update algorithm. And the private key that has not been updated can no longer decrypt the ciphertext.

**VI. SECURITY PROOF**

**A. CORRECTNESS**

The correctness of outsourcing decryption is shown below.

\[
E = \frac{\prod_{i \in I} e(C_i, K_2)}{e(C_1, K_1) \prod_{i \in I} e(D_i, K_{\rho(i)})} = e(g, g)^{\gamma s / u}.
\]

The correctness of decryption is shown below.

\[
C_0 \cdot E^{K_{u,2}}/e(C_1, K_{u,1}) = R \cdot e(g, g)^x \cdot e(g, g)^{\gamma s / u}.
\]

2) **SECURITY PROOF**

The first thing we should pay attention to is to prevent illegal users from colluding with D-SP. However, since each user’s master key is randomly divided, this collusion attack can be avoided. Specifically, suppose two users are requesting their private keys. TA randomly selects two sets of numbers \( (x_1, x_2) \) and \( (x_3, x_4) \), provided that \( x_1 + x_2 = x \mod p \) and \( x_3 + x_4 = x \mod p \) holds. Where \( x_1 \) and \( x_3 \) are used to generate private key components \( SK_{x_1} \) and \( SK_{x_2} \), and \( x_2 \) and \( x_4 \) are used to generate local private key components \( SK_{x_1} \) and \( SK_{x_2} \). The ciphertext can only be decrypted correctly if \( SK_{x_1} \) and \( SK_{x_2} \) match. Therefore, even if a group of curious users collude with D-SP to obtain all \( SK_{x_1} \), they cannot forge valid private keys to successfully perform decryption outside their scope.
Next, we prove that the security of the scheme can resist Type-II adversary.

**Theorem 1:** Assume that the $q$-DBDHE assumption holds in $G_1$ and $G_T$, then the advantage of all polynomial time adversaries who can win the IND-sCP-CPA game is negligible.

**proof:** Assume the advantage of distinguishing a valid ciphertext from a random element for a polynomial time adversary $A$ is $\epsilon_1 = \text{Adv}_{A}^{\text{IND-sCP-CPA}}$. There is a simulator $B$ who can break the $q$-DBDHE assumption with non-negligible advantage $\epsilon_1/2$.

Firstly, the $q$-DBDHE challenger $C$ sets

$$\Phi = (g, g^s, g^{a_1}, \ldots, g^{a_i}, \ldots, g^{a_{q+2}}, g^{ab_1}, g^{ab_2}/b_i, \ldots, g^{ab_{q+2}}/b_i, g^{ab_{q+3}}/b_i, \ldots, g^{ab_{q+2}/b_i})$$

where $a, s, b_1, \ldots, b_q$ are chosen in $Z_p$ at random and $k, j \in [1, q], k \neq j$.

Next, $C$ selects $\gamma \in \{0, 1\}$. If $\gamma = 1$, $C$ selects a random element $z$ in $G_T$. Otherwise, $T = e(g, g)^{q+1}$.

**Init:** $B$ is granted a $q$-DBDHE challenge instance $(\Phi, Z)$. $A$ announces a challenging access policy $(A^s, \rho^s)$, and transmits it to $B$.

**Setup:** The simulator $B$ selects a number $x'$ in $Z_p$ randomly, then sets $e(g, g)^x = e(g^{a_1}, g^{a_2}) \cdot e(g, g)^{x'}$ to make $x = x' + q + 1$ implicitly. To simulate elements $APK_\theta$, the simulator $B$ selects a random value $\nu_\theta$ for each attribute $\theta$ and sets $APK_\theta = g^{\nu_\theta}$. If $\rho^s(i) = \theta$ establishes, $B$ randomly selects $\nu_\theta \in Z_p$, and processes as:

$$H(\theta) = g^{q + \sum_{i \in J} g^{\rho^s(i)}M_{I_1}^{b_i} \cdot g^{\rho^s(i)}M_{I_2}^{b_i} \cdot \cdots \cdot g^{\rho^s(i)}M_{I_{q+2}}^{b_i}}$$

where $J$ means the set of indices $i$. If $J = \emptyset$ then $B$ sets $H(\theta) = g^s$, and the value of $H(\theta)$ is distributed at random owing to the value $g^s$. Finally, $B$ gives partial public parameters to $A$.

**Phase I:** $B$ builds a list of tuples $(S, SK)$ which is represented as $L_{SK}$ and empty initially. $A$ can make any inquiries as follows:

- **$O_{SK}(S)$:** Assume that $A$ gives private key inquiries about attribute set $S$ which can not match $(A^s, \rho^s)$, $B$ performs the following operations.

If $A$ has previously asked for $S$, $B$ retrieves $SK$ from the list $L_{SK}$ and sends it to $A$; Otherwise, $B$ selects a vector $Y = (Y_1, \ldots, Y_{K^r}) \in Z_p$ with $Y_1 = -1$ and $A^s \cdot Y = 0$ for every $i$ where $\rho(i) \in S$. $B$ selects $x'_i, x'_2 \in Z_p$ randomly such that $x'_i + x'_2 = x' \mod p$, and sets $x_1 = x'_1 + q + 1, x_2 = x'_2$. Then $B$ calculates $K_{t,1}$ as:

$$K_{t,1} = g^{x'_1} = g^{x_1}.$$  

$B$ picks an element $\sigma \in Z_p$ randomly, then denotes $t$ implicitly as

$$t = \sigma + \sum_{i \in J} \rho^s(i)\cdot Y_i \cdot g^{\rho^s(i)}.$$  

Then $B$ can calculate $K_{t,2}$ as:

$$K_{t,2} = g^{x'_2} \cdot \prod_{i=1,\ldots,n^*} (g^{\rho^s(i)}Y_i) = g^{x/u}.$$  

We can see that $g^{x/u}$ contains the following forms $g^{x + i}$ from the definition of $t$, which can be ignored with the unknown terms in $g^{x + i}$ when calculating $K_{t,1}$. $B$ computes

$$K_{t,1} = g^{x'} \cdot \prod_{i=2,\ldots,n^*} (g^{\rho^s(i)}Y_i) = g^{x'/u}.$$  

Next $B$ denotes $K_{t,2}$ for each $\theta \in S$. If $i$ does not exist such that $\rho^s(i) = \theta$ for each attribute $\theta \in S$, let $K_{t,2} = g^{x'/u}$.

However, $B$ can’t simulate $K_{t,2}$ for the attribute $\theta \in S$ used in $A^s$ since $K_{t,1}$ includes terms with the form $g^{x + i}$. But $B$ has $A^s \cdot Y = 0$ so that all terms of the form $g^{x + i}$ are canceled. $B$ sets $K_{t,2}$ as:

$$K_{t,2} = g^{x'/u}.$$  

$B$ puts the secret key $SK$ into list $L_{SK}$ and sends it to $A$.

- **$O_{UK}(\theta)$:** $A$ sends a canceled attribute $\theta'$ to make the updated attribute version key query. $B$ selects a new random number $\nu_\theta' \in Z_p$ for the canceled attribute $\theta'$, and sets the updated attribute version key $AVK_{\theta'} = \nu_\theta' \cdot \nu_\theta$, then returns it to $A$.

**Challenge:** $A$ selects two equal-length message $R_0, R_1$ to $B$. Then $B$ picks $r \in \{0, 1\}$ randomly, and sets $C_r^i$ below.

$$C_r^i = R_r \cdot T \cdot e(g, g)^{x'/u}.$$  

Since the component $C_r^i$ includes terms $g^{\rho(i)}$, how to simulate $C_r^i$ for $B$ is difficult. To overcome the difficulty, $B$ selects $y_2', \ldots, y_{n^*} \in Z_p$ randomly, then distributes $s$ through the following vector

$$v = (x, sa + y_2, sa^2 + y_3, \ldots, sa^{n^*} - 1 + y_{n^*}) \in \mathbb{Z}_p^{n^*}.$$  

Let $Q_k$ be the set for all $i \neq i$ which satisfies $\rho(i) = \rho(i) \in [1, l]$. $B$ also picks a series of random numbers $r'_1, \ldots, r'_l \in Z_p$. $B$ sets $D_{r'}^i$ and $C_{r'}^i$ by setting $r_i = r'_i - \nu_{b_i}$ implicitly as:

$$D_{r'}^i = g^{x'_i - \nu_{b_i} + r'_i} \cdot \prod_{i=2,\ldots,n^*} (g^{\rho(i)}M_{I_x}^{b_i} \cdot Y_i).$$  

Then $B$ passes the challenge ciphertext $CT^* = \{C^i_0, C_r^i, C_{r'}^i, D_{r'}^i\}_{i=1,\ldots,l}$ to $A$.  

104352  

VOLUME 8, 2020
Phase II: A makes queries similar to phase I, and B answers like phase I.

Guess: A gives a guess $r'$. If $r' = r$, B will output $\gamma' = 1$ to indicate that $Z$ is random element in group $G_T$. Otherwise, $B$ will output $\gamma' = 0$ showing that $Z = e(g, g)^{a(1+\epsilon)}$. If $Z = e(g, g)^{a(1+\epsilon)}$, it shows that $A$ will win the game with a non-negligible $\epsilon_1$ by definition above. And $B$’s advantage of solving the $q$-DBDHE problem is $Adv_B = 1/2 + \epsilon_1$. Otherwise $Adv_B = 1/2$. Hence, the $B$’s advantage of solving the $q$-DBDHE problem is computed as follows.

$$Adv_B = \frac{1}{2} \left( \frac{1}{2} + \epsilon_1 \right) + \frac{1}{2} \left( \frac{1}{2} - 1 \right) - \frac{1}{2} = \frac{\epsilon_1}{2}$$

Theorem 2: The proposed scheme is secure against selectively chosen keyword attack based on the given one-way hash function $H$.

Proof: In the interactive game below, $A$ aims to distinguish $A$ from $B$. Selecting a random value $f \in Z_p$, the advantage of $A$ in distinguishing $g_f$ from $g_{(a(1+\epsilon))}^{\phi(H(w))}$ is the same as that of differentiating between $g_f$ and $g_{(a(1+\epsilon))}^{\phi(H(w))}$. For brevity, assume that $A$ can distinguish $g_f$ from $g_{a(1+\epsilon)}$. Then we define an secure interactive game between $A$ and $B$ below.

**Setup.** $B$ selects $a, x \in Z_p$ randomly, then returns public keys $PK = (g, g^a, g^x)$ to $A$. And $B$ owns $MSK = \{x\}$.

**Phase I.** $A$ can inquire the $O_{SK}$ and $O_{TD}$ for the secret key and query trapdoor in polynomial time.

- $O_{SK}(PK, MSK):$ B calculates $K' = g^{a_2}$, then transmits $K'$ to $A$.
- $O_{TD}(PK, w^*, K'): $ Inputting $PK, w^*$ and $K'$, $B$ randomly selects $a \in Z_p$ and generates a trapdoor $TD_a = (T_1, T_2, T_3)$ of the query keyword $w^*$, where $T_1 = g^{a\alpha} g^{H(w)}$, $T_2 = g^{a_2x}$, $T_3 = g^{a\alpha x}$.

**Challenge.** $A$ inputs two same-length keywords $w_0$ and $w_1$ which were not used for trapdoor query before. $B$ selects $\xi, \pi \in Z_p$, and picks $r \in \{0, 1\}$. If $r = 0$, $B$ sets $I_1 = g^{\xi + \epsilon}, I_2 = g^\xi, I_3 = g^\epsilon$, otherwise, $B$ sets $I_1 = g^{\xi + \epsilon}, I_2 = g^{(a(1+\epsilon))}, I_3 = g^\epsilon$.

**Phase II.** $A$ makes queries similar to phase I with the restriction $w \neq w_0, w_1$, and $B$ answers like phase I.

If $A$ obtains $e(g, g)^{\phi(H(\xi + \epsilon))}$ with the term $g^\epsilon$ returned by the query, then $A$ can distinguish $g^{(a(1+\epsilon))}$ from $g^\epsilon$. But we also need to prove that the $A$’s advantage of obtaining $e(g, g)^{\phi(H(\xi + \epsilon))}$ with the term $g^\epsilon$ is negligible in the game.

Let $G_1 = \{\phi_1(n) | n \in Z_p\}, G_T = \{\phi_2(n) | n \in Z_p\}$, where $\phi_1, \phi_2$ are two injective functions which map from $Z_p$ to $\{0, 1\}^\theta$, with $\theta > 3\log(p)$. In the mapping between $\phi_1$ and $\phi_2$, the advantage of distinguishing elements can be negligible. We will discuss the $A$’s advantage in building $e(g, g)^{\phi(H(\xi + \epsilon))}$ with the term $g^\epsilon$.

Firstly, we think about that how to obtain $e(g, g)^{\phi(H(\xi + \epsilon))}$ from $g^\epsilon$. Since only the item $x\xi$ contains the $\xi$, the item $h'$ must include $\epsilon$ to obtain $e(g, g)^{\phi(H(\xi + \epsilon))}$. $A$ will try to build $e(g, g)^{\phi(H(\xi + \epsilon))}$ based on $h' = h^\epsilon \bar{\xi}$. But $A$ also needs to obtain $h^\epsilon \bar{\xi} \alpha \tau$ with the terms $\tau$ and $\alpha \bar{x}$. Since only $B$ owns the master key $\bar{x}$, $A$ cannot get $e(g, g)^{\phi(H(\xi + \epsilon))}$ anyway.

Finally, we can see that $A$ cannot distinguish $g^{(a(1+\epsilon))}$ from $g^\epsilon$. So $A$ cannot distinguish between $g^{(a(1+\epsilon))} g^{\xi} H(w_0)$ and $g^{(a(1+\epsilon))} g^{\xi} H(w_1)$. That is to say, $A$ cannot break the proposed scheme with a non-negligible advantage.

**VII. PERFORMANCE ANALYSIS**

**A. FUNCTIONAL COMPARISON**

The proposed scheme has the following functions: keyword search, result ranking, decryption verification, attribute revocation. Table 2 provides the functional comparison between our scheme and other existing schemes, [23], [26], [36].

**B. THEORETICAL ANALYSIS**

In this subsection, we compared our scheme with some existing schemes. $E$ means the modular exponentiation operation in group $G_1$ and $E_T$ represents the modular exponentiation operation in $G_T$. $P$ means the pairing operation, $H$ means hash operation. Moreover, for comparing the storage overhead of different schemes, we use $|G_1|, |G_T|, |Z_p|$ to represent the size of elements in $G_1, G_T, Z_p$ respectively. And $|PK|, |MSK|, |SK|, |CT|, |TD|$ mean the size of system public keys, master secret key, private key, ciphertext and trapdoor, respectively. The comparison of different schemes can be seen in Table 3 and Table 4.
TABLE 4. Storage comparison of schemes.

| Scheme     | Storage cost comparison |
|------------|-------------------------|
| ABEKSr [23] | $\lceil n + 3 \rceil | G_1 \rceil + | G_7 \rceil | |
| MSK        | $(n + 2) | Z_p \rceil | |
| SK         | $\frac{(2y + 1)}{2} | G_1 \rceil + | G_7 \rceil | |
| CT         | $\frac{(2t + 2)}{2} | G_1 \rceil + | G_7 \rceil | |
| TD         | $\frac{2}{2} | G_1 \rceil + | G_7 \rceil | |
| VKSE [36]  | $\lceil 5 \rceil | G_1 \rceil + | G_7 \rceil | |
| RSABE [6]  | $(3n + 1) | G_1 \rceil + | G_7 \rceil | |
| Ours       | $(n + 3) | G_1 \rceil + | G_7 \rceil | |

**C. EXPERIMENTAL ANALYSIS**

It is important to note that we separately illustrate the overhead generated by the index generation process. In the scheme, DO establishes keyword indexes, which consists of two parts: index table and many query arrays. The calculation complexity of index table is $O(m)$, where $m$ means the number of keywords. Each query array stores all the file identifiers containing corresponding keywords, so the calculation cost of generating each array is different. The computational complexity of generating all arrays is $O(F)$, where $F = \sum_{\tau=1}^{m} |F(w_\tau)|$ represents the number of all files which are associated with the certain keyword $w_\tau$. Hence, the time cost in the index generation phase depends on the size of the dictionary and the file list $F(w_\tau)$ containing the corresponding keywords.

Besides, we compared the computational overhead with other existing schemes in the phases of system initialization, private key generation, trap generation and search. Figure 4 (a) indicates the computational cost of our scheme in setup phase is lower than that of the ABKS-UR scheme. From Figure 4 (b), the computing overhead of the scheme increases linearly with the increase of the number of attributes in KenGen phase, but the computational overhead of the scheme is lower under the same number of attributes. From figure 4(c) and 4(d), the computational overhead of the VKSE and ABKS-UR schemes increases linearly with the number of attributes in TrapdoorGen stage and search stage respectively, but the computational overhead of our scheme remains at a low level.

As for the comparison of storage cost of the scheme, we command $|G| = 1024$ bit and $|Z_p| = 160$ bit in the actual simulation environment. The comparison results are shown in Figure 5.

Besides, we can see that the storage overhead of our scheme is lower than that of the ABKS-UR scheme in the generation of system public parameters, master keys and user secret keys from Figure 5(a), (b), and (c). Besides, Figure 5 (d) shows that in the ABKS-UR scheme, the storage overhead of the trapdoor
increases linearly with the number of user attributes, while the storage overhead of the trapdoor remains at a low level in our scheme.

VIII. CONCLUSION

Although searchable encryption technology allows users to retrieve ciphertext without decryption, not all query results are within interest for DU. Therefore, this paper proposes a ranked keyword search scheme that can be based on the keyword of the query. Ranking the relevant scores of keywords and files and returning the top-k files, which greatly saves network bandwidth and resource consumption, and improves the user’s retrieval experience. The proposed scheme also supports attribute revocation. In the end, we show the proposed scheme achieves IND-sCP-CPA security and IND-CKA security.

In order to further improve the accuracy of the search, we consider that in the future we will extend the ranking function to a multi-keyword system with the same features as the proposed scheme. However, when the user submits an interested keyword sets, they cannot guarantee that each file contains these keywords. Therefore, how to apply ranking function to multi-keyword search is a difficulty, and our next stage will focus on this.

REFERENCES

[1] X. Wang, Z. Ning, M. Zhou, X. Hu, L. Wang, Y. Zhang, F. R. Yu, and B. Hu, “Privacy-preserving content dissemination for vehicular social networks: Challenges and solutions,” *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1314–1345, 2nd Quart., 2019.

[2] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization,” in *Public Key Cryptography*, D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, Eds. Berlin, Germany: Springer, 2011, pp. 53–70.

[3] M. H. Gahramani, M. Zhou, and C. T. Hon, “Toward cloud computing QoS architecture: Analysis of cloud systems and cloud services,” *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 1, pp. 6–18, Jan. 2017.

[4] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2000, pp. 44–55.

[5] B. Dan, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in *Advances in Cryptology*. Berlin, Germany: Springer, 2004, pp. 506–522.

[6] 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 Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 1187–1198, Apr. 2016.

[7] J. Han, Y. Yang, J. K. Liu, J. Li, K. Liang, and J. Shen, “Expressive attribute-based keyword search with constant-size ciphertext,” *Soft Comput.*, vol. 22, no. 15, pp. 5163–5177, Aug. 2018.

[8] Y. Miao, Q. Tong, K.-K.-R. Choo, X. Liu, R. H. Deng, and H. Li, “Secure online/offline data sharing framework for cloud-assisted industrial Internet of Things,” *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8681–8691, Oct. 2019.

[9] B. D. Revathy, A. Anbumani, and M. P. Avishankar, “Enabling secure and efficient keyword ranked search over encrypted data in the cloud,” *Int. J. Recent Adv. Sci. Eng.*, vol. 4, no. 2, pp. 28–32, 2015.

[10] X. Jiang, J. Yu, J. Yan, and R. Hao, “Enabling efficient and verifiable multi-keyword ranked search over encrypted cloud data,” *Inf. Sci.*, vols. 403–404, pp. 22–41, Sep. 2017. doi: 10.1016/j.ins.2017.03.037.

[11] Y. Liu, H. Peng, and J. Wang, “Verifiable diversity ranking search over encrypted outsourced data,” *Comput. Mater. Continua*, vol. 55, no. 1, pp. 37–57, 2018.

[12] J. Li, W. Yao, Y. Zhang, H. Qian, and J. Han, “Flexible and fine-grained attribute-based data storage in cloud computing,” *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 785–796, Sep./Oct. 2017.
H. Li, D. Liu, Y. Dai, T. H. Luan, and S. Yu, “KSF-OABE: Outourced attribute-based encryption with keyword search function for cloud storage,” IEEE Trans. Services Comput., vol. 10, no. 5, pp. 715–725, Sep./Oct. 2017, doi: 10.1109/TSC.2016.2542813.

S. Qiu, J. Liu, Y. Shi, and R. Zhang, “Hidden policy ciphertext-policy attribute-based encryption with keyword search against keyword guessing attack,” Sci. China Inf. Sci., vol. 60, no. 5, May 2017, Art. no. 052105, doi: 10.1007/s11432-015-5449-9.

J. Sun, L. Ren, S. Wang, and X. Yao, “Multi-keyword searchable and data verifiable attribute-based encryption scheme for cloud storage,” IEEE Access, vol. 7, pp. 66655–66667, 2019.

H. Li, D. Liu, Y. Dai, T. H. Luan, and S. Yu, “Personalized search over encrypted data with efficient and secure updates in mobile clouds,” IEEE Trans. Emerg. Topics Comput., vol. 6, no. 1, pp. 97–109, Jan./Mar. 2018.

A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in Advances in Cryptology, R. Cramer, Ed. Berlin, Germany: Springer, 2005, pp. 457–473.

D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing,” SIAM J. Comput., vol. 32, no. 3, pp. 586–615, Jun. 2003.

V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in Proc. 13th ACM Conf. Comput. Commun. Secur. (CCS), 2006, pp. 89–98.

J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in Proc. IEEE Symp. Secur. Privacy, May 2007, pp. 321–334.

M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, “Secure attribute-based systems,” in Proc. 13th ACM Conf. Comput. Commun. Secur. (CCS), New York, NY, USA, 2006, pp. 99–112.

L. Ibráimí, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, “Mediated ciphertext-policy attribute-based encryption and its application,” in Proc. 10th Int. Workshop Inf. Secur. Appl. (WISA), in Lecture Notes in Computer Science, vol. 5932, H. Y. Youm and M. Yung, Eds. Berlin, Germany: Springer, 2009, pp. 309–323.

J. Wang, X. Yin, J. Ning, and G. S. Poh, “Attribute-based encryption with efficient keyword search and user revocation,” in Proc. Int. Conf. Inf. Secur. Cryptol. Cham, Switzerland: Springer, 2018, pp. 490–509.

J. Hur and D. K. Noh, “Attribute-based access control with efficient revocation in data outsourcing systems,” IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 7, pp. 1214–1221, Jul. 2011.

Z. Liu, Z. L. Jiang, X. Wang, and S. M. Yiu, “Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating,” J. Netw. Comput. Appl., vol. 108, pp. 112–123, Apr. 2018.

S. Wang, D. Zhang, Y. Zhang, and L. Liu, “ Efficiently revocable and searchable attribute-based encryption scheme for mobile cloud storage,” IEEE Access, vol. 6, pp. 30444–30457, 2018.

Z. Li, W. Li, Z. Jin, H. Zhang, and Q. Wen, “An efficient ABPE scheme with verifiable outsourced encryption and decryption,” IEEE Access, vol. 7, pp. 29023–29037, 2019.

J. Li, Y. Wang, Y. Zhang, and J. Han, “Full verifiability for outsourced decryption in attribute based encryption,” IEEE Trans. Services Comput., early access, May 31, 2017, doi: 10.1109/TSC.2017.2710190.

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, Feb. 2016.

L. Xue, Y. Yu, Y. Li, M. H. Au, X. Du, and B. Yang, “Efficient attribute-based encryption with attribute revocation for assured data deletion,” Inf. Sci., vol. 479, pp. 640–650, Apr. 2019, doi: 10.1016/j.ins.2018.02.015.

P. Zhang and M. Zhou, “Dynamic cloud task scheduling based on a two-stage strategy,” IEEE Trans. Autom. Sci. Eng., vol. 15, no. 2, pp. 772–783, Apr. 2018.

H. Zhu, Z. Mei, B. Wu, H. Li, and Z. Cui, “Fuzzy keyword search and access control over ciphertexts in cloud computing,” in Proc. Australas. Conf. Inf. Secur. Privacy, 2017, pp. 248–265.

Y. Zhang, A. Wu, T. Zhang, and D. Zheng, “Secure and flexible keyword search over encrypted data with outsourced decryption in Internet of Things,” Ann. Telecommun., vol. 74, nos. 7–8, pp. 413–421, Aug. 2019, doi: 10.1007/s12243-018-0694-8.

P. Xu, S. He, W. Wang, W. Sasilo, and H. Jin, “Lightweight searchable public-key encryption for cloud-assisted wireless sensor networks,” IEEE Trans. Ind. Informat., vol. 14, no. 8, pp. 3712–3723, Aug. 2018.

Y. Zhang, D. Zheng, and J. Li, “Attribute directly-reversible attribute-based encryption with constant ciphertext length,” J. Cryptol. Res., vol. 1, no. 5, pp. 465–480, 2014.

Y. Miao, J. Ma, Q. Jiang, X. Li, and A. K. Sangahaa, “Verifiable keyword search over encrypted cloud data in smart city,” Comput. Electr. Eng., vol. 65, pp. 90–101, Jan. 2020.

J. Li, Y. Zhang, J. Ning, X. Huang, G. S. Poh, and D. Wang, “Attribute based encryption with privacy protection and accountability for CloudIoT,” IEEE Trans. Cloud Comput., early access, Feb. 19, 2020, doi: 10.1109/TCC.2020.2975184.

H. Zahid, T. Mahmood, A. Marshed, and T. Sellis, “Big data analytics in telecommunications: Literature review and architecture recommendations,” IEEE/CAA J. Autom. Sinica, vol. 7, no. 1, pp. 18–38, Jan. 2020.

Y. Luo, M. Xu, K. Huang, D. Wang, and S. Fu, “Efficient auditing for shared data in the cloud with secure user revocation and computations outsourcing,” Comput. Secur., vol. 73, pp. 492–506, Mar. 2018.

J. Li, W. Yao, J. Han, Y. Zhang, and J. Shen, “User collusion avoidance CP-ABE with efficient attribute revocation for cloud storage,” IEEE Syst. J., vol. 12, no. 2, pp. 1767–1777, Jun. 2018.

**LEYOU ZHANG** received the M.S. and Ph.D. degrees from Xidian University, in 2002 and 2009, respectively. He is currently a Professor with the School of Mathematics and Statistics, Xidian University. His current research interests include computing security, network security, and cryptography.

**JIAN SU** received the B.S. degree in mathematics from Shandong Jianzhu University, in 2017. He is currently pursuing the M.S. degree in applied mathematics with Xidian University. His current research interests include applied cryptography, ranked keyword search, and information security.

**YI MU (Senior Member, IEEE)** received the Ph.D. degree from Australian National University, Canberra, ACT, Australia, in 1994. He was a Professor of computer science with the University of Wollongong, Wollongong, NSW, Australia, in 2018. He is currently a Professor with the College of Mathematics and Informatics, Fujian Normal University, Fuzhou, China. His current research interests include blockchain, cybersecurity, and cryptography. He was the Editor-in-Chief of the International Journal of Applied Cryptography. He has served as an associate editor for several other international journals.