Ciphertext Query Method Based on Multi-Attribute Keywords

Haoyu Li, Longjun Zhang, Qingpeng Li and Zhiqiang Gao
Department of Information Engineering, Engineering College of the Chinese Armed Police Force, Xi'an, Shaanxi, China

***Abstract*—Following the widespread application of cloud computing, many users outsource local data to cloud server. Users encrypt their data before outsourcing makes the traditional query method failure. Aiming at the problem of slow query speed and low accuracy of ciphertext query, an improved ciphertext query method based on multi-attribute keywords is proposed through the research of existing ciphertext query technique. The method establishes security index through multi-attribute characteristic vector, it conducts correlation calculation according to the sort order selected by users and multi-keyword, and returns the most relevant sorted query result. Research result shows that the method improves query speed and accuracy of the query result.**

Keywords-cloud computing; ciphertext search; multi-attribute keywords

I. INTRODUCTION

With the development of information technology, many fields have produced large amounts of sensitive data, improve access speed and data security has become an urgent problem. The widespread application of cloud computing provides ideas to solve the problem. Users can get high quality service only spend a little money, which make more institutions store their local data in the cloud, the cloud service providers (CSP) responsible for the unified management of preservation. Data security received extensive attention, the data stored in the cloud are vulnerable to external network attacks and threat by the honest-but-curious [1] administrator from the CSP. Encrypt data locally and then upload to the cloud are one of the solutions to the security threat [2]. But encrypt data upload make the traditional retrieval mechanism failure, how to retrieve the encrypted data quickly and efficiently has become a huge challenge.

II. RELATED WORK

In recent years, query ciphertext data raised a lot of ways, the common way is keyword-based method, the method can meet users' need for they only query what they care about. At present, there have two kinds of query methods based on keyword, one is based on single keyword and the other one is based on multi-keywords. In the study of single keyword query method, Song et al [3] put forward single keyword ciphertext query method based on symmetric key first, then Goh et al [4] improved the efficiency of the method, Wang et al [5] proposed ciphertext query method that support result sorting. However, these methods cannot satisfy the needs of user's personalized query. Many researchers began to study the ciphertext query based on multi-keywords, Cao et al [6] proposed the ciphertext sorting method based on multi-keywords, implement the result of sorting functions. Sun et al [7] proposed a new ciphertext query framework MRSE to solve the ciphertext indexing problems, but the query time increase rapidly. Tian et al [8] proposed a new structure based on similarity search tree, but not mention about how to sort the data. Many existing methods cannot meet the requirement of the ciphertext query in cloud environment. In order to improve query speed and accuracy, we propose an improved ciphertext query method based on multi-attribute keywords, according to the user's choice of sorting method and multi-keywords to do the correlation calculation, return the most relevant result.

III. CIPHERTEXT QUERY METHOD

The existing methods spend too much time and the accuracy is not high, how to discover ciphertext data quickly and accurately in the cloud environment is a big challenge. So we propose an effective method to make up for these shortcomings.

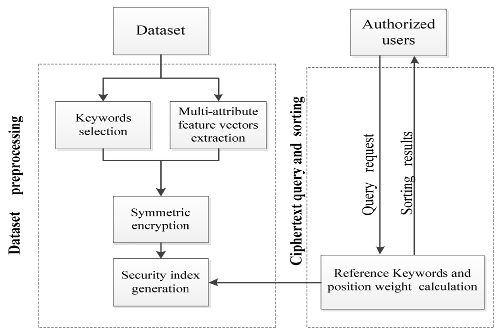
Dataset
Authorized users
Dataset preprocessing
Keywords selection
Multi-attribute feature vectors extraction
Symmetric encryption
Security index generation
Ciphertext query and sorting
Query request
Sorting results
Reference Keywords and position weight calculation

FIGURE I. PROCESS OF MULTI-ATTRIBUTE KEYWORDS CIPHERTEXT QUERY

Figure I illustrate the process of the ciphertext query method. It consists of two parts: data preprocessing, ciphertext query and sorting.

A. *Keywords Selection*

Input the document and use segmentation algorithm to separate the words in the document, form a result set, then process the result set by calculating the words frequency and filter words, finally extract the keywords which can accurately reflect the meaning of the document.

We use IF-IWF [9] (Term Frequency-Inverse Word Frequency) algorithm to extract the keywords, formula is as follows:

$$TF_{i,j} \times IWF_i = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{\sum_{i=1}^{m} nt_i}{nt_i} \tag{1}$$

The first half of the molecules: $n_{i,j}$ represents the number of the word t_i appear in the document, the denominator represents the sum of the frequency of all words in the document. Second half of the molecule represents the number of all words in the dictionary, the denominator represents the sum of the frequency of the word t_i. This method reduces the effect of the same type of document weight for words, make the results more accurate. After the completion of the calculation, we can get the keywords set K= {k_1, k_2,…, kn}.

B. Multi-Attribute Feature Vectors Extraction

Before extract the multi-attribute feature vector. We should define some basic concepts.

- Partial attribute of the keywords: keywords are affected by partial factors of the document.
- Global attribute of the keywords: keywords are affected by global factors of the document.
- Partial attribute of the feature vectors: The feature vectors that use partial attribute of the keywords.
- Global attribute of the feature vectors: The feature vectors that use global attribute of the keywords.
- Single attribute of the feature vectors: the keywords of feature vector have one attribute.

The vector space model (VSM) [10] is a typical application of the single attribute of the feature vectors, we can use a matrix to represent many documents in the vector spaces.

$$\begin{pmatrix} & C_1 & C_2 & \cdots & C_t \\ D_1 & d_{11} & d_{12} & \cdots & d_{1t} \\ D_2 & d_{21} & d_{22} & \cdots & d_{2t} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ D_n & d_{n1} & d_{n2} & \cdots & d_{nt} \end{pmatrix}$$

FIGURE II. SINGLE ATTRIBUTE OF THE FEATURE VECTORS

As shown in Figure II: D_1, D_2,…, D_n represent the number of the documents, C_1,C_2,…, C_t represent the characteristics of the document content, a matrix element corresponding to a word weight of the document.

- Multiple attribute of the feature vectors: the keywords of feature vector have many attributes.

1) Partial attribute feature vectors extraction

First, we extract partial attribute of the feature vectors of the document on the client side and use the keywords weight and keywords position as the partial attributes. If the user input keywords appear more frequently in the document, the position of the keywords appears more important, we will assume that the correlation between the keywords and document is much higher, the query result will be more accurate.

a) Keywords weight

We use TF-IDF [11] algorithm to calculate the keywords weight. Formula is as follows:

$$w_i = \frac{TF(t_i) \times IDF(t_i)}{\sqrt{\sum_{i=1}^{n} (TF(t_i) \times IDF(t_i))^2}} = \frac{TF(t_i) \times \log(\frac{N}{n_i} + 1)}{\sqrt{\sum_{i=1}^{n} (TF(t_i) \times \log(\frac{N}{n_i} + 1))^2}} \tag{2}$$

w_i is the weight of the keywords, TF represents the number of times that the keywords appear in the document, IDF represents the inverse document frequency, t_i represents the keywords, N represents the number of words in the dictionary, n_i represents the number of the document that include the keywords.

b) Keywords position weight

In general, queried keywords are likely to appear in the text. If the keywords appear in other locations, we need to determine the weight of the keywords according to their locations. If the keywords appear in the title of the document, it is also appear in the body of the text. The title can be considered as a summary of the text. So we give different weights according to the position of keywords. Shown in the following table.

TABLE I. KEYWORDS POSITION WEIGHT

Keywords Position	Weight
Title	6
The beginning of each paragraph	3
The end of each paragraph	3
Other locations	1

Next, we can extract the partial attribute feature vectors according to the formulas and table above. Calculate the words frequency weight fre_w and keywords position weight pos_w respectively from the keywords set K={ k_1,k_2,…, k_n}, then we can get the partial attribute feature vector

$$\vec{V_l} = \{(k_1, fre\_w, pos\_w), (k_2, fre\_w, pos\_w), \ldots , (k_n, fre\_w, pos\_w)\}.$$

2) Global attribute feature vectors extraction

Similar to the partial attribute feature vectors extraction, we regards downloaded times and cited times as the global attributes for they can reflect the attributes of the document as a whole, avoid totally relying on keywords attribute and reduce the retrieval relevance.

a) Download times

It represents the number of users to download the document, we can realize that how many times the document has been consulted. Download times recorded by the cloud server.

b) Cited times

It represents the number of users to cite the document and reflects the quality of the document from the side.

According to these two attributes, the cloud server will generate the global attribute of the feature vector $\vec{V}_g$.

C. Symmetric Encryption

Assuming that the cloud server is not credible, we need to encrypt the document on the client side to ensure data security. Due to the high efficiency of the symmetric encryption algorithm and it can meet the requirements of general users of security, we choose the symmetric encryption algorithm to encrypt the document. Only keywords part of the partial attribute feature vectors may reveal the document information, so we use trapdoor algorithm to encrypt them, then we will get the partial attribute feature vector $\vec{E}_{V_l}$ = $\{(Ek_1,\ fre\_w,\ pos\_w),\ (Ek_2,\ fre\_w, pos\_w),\ldots,(Ek_n,\ fre\_w, pos\_w)\}$. Finally, upload the encrypted document and partial attribute feature vector to the cloud server.

D. Security Index Generation

Establish security index is a key step of the method. Inverted index [12] is a common method, it finds data based on attribute weight and consists of two parts: keywords dictionary and inverted list. Keywords dictionary consists of the uploaded keywords, inverted list records all document list related to the keywords and the position information of the keywords. The structure of inverted index shown as below.

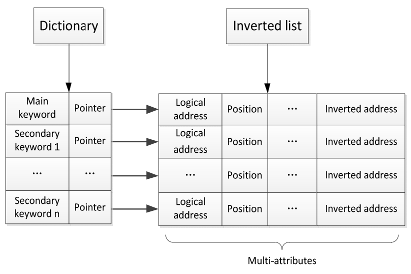

FIGURE III. INVERTED INDEX STRUCTURE

The cloud server will extract the global attribute feature vector $\vec{V}_g$ when it receives the encrypted document from the user. Combine $\vec{V}_g$ and $\vec{E}_{V_l}$, we will get the encrypted document with multiple attribute feature vector $\vec{E}_{V_m} = \{(Ek_1,\ fre\_w,\ pos\_w, d\_num, r\_num), (Ek_2,\ fre\_w,\ pos\_w, d\_num, r\_num),\ldots,(Ek_n,\ fre\_w,\ pos\_w, d\_num, r\_num)\}$. According to the document number and multiple attribute feature vector establish the ciphertext inverted index.

E. Ciphertext Retrieval and Relevance Sorting

This is the final step, authorized users input keywords to search the ciphertext, after the cloud server receives the request, it will calculate the keyword weight and evaluate multiple attributes, return the sorting result eventually.

We deal with the user's query request first, user input with or expression (e.g. $Q = w_1 \cup w_2 \cup \ldots \cap w_i$) consists of main keyword K1 and several secondary keywords K2,…,Kn, hope to get satisfy the logical expression of all the documents. Trapdoor algorithm produces the trapdoor value $T_w = \{Tk_1, Tk_2, \ldots,\ Tk_n\}$ about these keywords.

Then the user can choose different ways to sort, such as topic sort, published time sort, download times sort and so on. In a different sort of way, the weights of different attributes are also different, but the main keyword frequency weight should possess the highest proportion in any way of sorting.

Table II shows the different way of sorting decide that the different attribute weights are different. After the user input keywords and select the sorting way, they will submit the trapdoor value of the keywords and sorting method to the cloud server.

TABLE II. ATTRIBUTE WEIGHTS OF DIFFERENT WAYS OF SORTING

Sort of way	**Weight name**				
	Main keyword	***Secondary keywords***	***position***	***Download times***	***Cited times***
Topic	0.4	0.1	0.3	0.1	0.1
Cited times	0.3	0.1	0.15	0.15	0.3
Download times	0.3	0.1	0.15	0.3	0.15

The cloud server retrieves the ciphertext index table according to the trapdoor value submitted by the user and lists preliminary results, then calls the multi-attribute evaluation function calculated according to the different ways of sorting, optimize the sorting results and return the best result to the user.

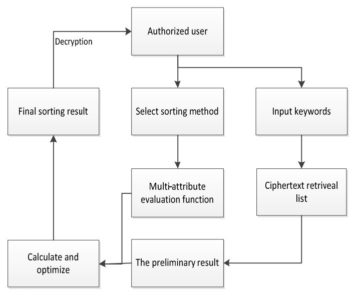

FIGURE IV. CIPHERTEXT RETRIEVAL AND RELEVANCE SORTING

Figure IV illustrates the specific process of the ciphertext retrieval and relevance sorting. Multi-attribute evaluation function uses partial attribute and global attribute to calculate the relevance of documents. Assume that the main keyword and several secondary keywords $\beta_1, \beta_2, \ldots, \beta_n$ corresponding weight of $\delta_1, \delta_2, \ldots, \delta_n$, $\sum_{i=1}^{n} \delta_i = 1$, Multi-attribute evaluation function formula is as follows:

$$F(x) = \sum_{i=1}^{n} \delta_i \times \beta_i \tag{3}$$

IV. Performance & Evalution

In this section, we introduce the experimental environment first. Hardware environment: Intel Core i3-2330M 2.20 GHz processor, 4G memory. The operating system is Windows 7 64bit. Language development environment: MyEclipse9.1, JDK1.7, Lucene-3.6.2.

A. Performance Evaluation Index

We use query accuracy rate and recall rate as performance evaluation indexes.

1) Query accuracy rate formula:

$$\text{Accuracy rate}\left(\text{P}\right)=\frac{\text{Relevant documents}}{\text{All documents}} \tag{4}$$

2) Recall rate formula:

$$\text{Recall rate}\left(\text{R}\right)=\frac{\text{Relevant documents}}{\text{All relevant documents}} \tag{5}$$

B. Experiment Analysis

We randomly selected 100 articles in the paper library, contrasted the single attribute ciphertext query time with multi-attribute ciphertext query time and also analyzed the document relevancy.

1) Query time analysis

We input the same keywords in single attribute ciphertext query system and multi-attribute ciphertext query system, return in the top 20,40,60,80,100 results with query time.

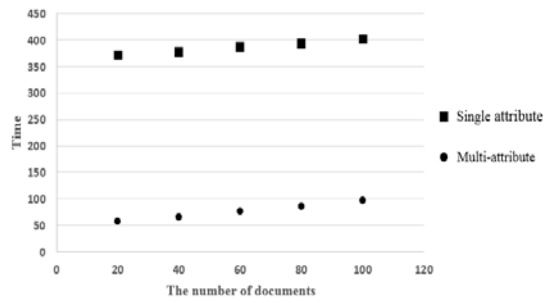

FIGURE V. QUERY TIME COMPARISON

As shown in Figure V, both two query system time increase with the increase of the number of articles. We can also find that in the same number of articles, multi-attribute ciphertext query system spend less time.

2) Document relevancy analysis

We input a different number of keywords in single attribute ciphertext query system and multi-attribute ciphertext query system, return correlation of the document.

As shown in Figure VI, multi-attribute ciphertext query system returned more relevant articles, the query results more accurate.

These two experiments show that the ciphertext query method we proposed can improve the Q efficiency and the relevance of the document.

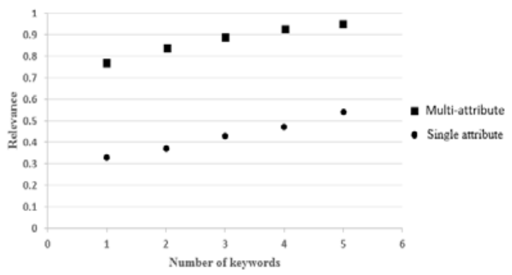

FIGURE VI. RELEVANCE DOCUMENT COMPARISON

V. Conclusions

In this paper, in order to improve efficiency of the ciphertext retrieval, we have presented a ciphertext query method based on multi-attribute keywords. We use multi-attribute feature vector to establish inverted index and calculate the correlation of documents by multi-attribute evaluation function, eventually return the most relevant articles and sort them. The experimental results show that the method can effectively improve the ciphertext query speed and the accuracy of the query results.

References

[1] H. Hacigumus, B. Iyer, S. Mehrotra, Providing Database as a Service [C], Proceedings of the 18th International Conference on Data Engineering. IEEE Computer Society, 2002:29-38.

[2] S. Kamara, K. Lauter, Cryptographic Cloud Storage [J]. Financial Cryptography & Data Security, 2010:136-149.

[3] DX. Song, D. Wagner, A. Perrig, Practical Techniques for Searches on Encrypted Data [J]. IEEE Symposium on Security & Privacy, 2012:44-55.

[4] Goh E J. Secure Indexes [J], Submission, 2004.

[5] C. Wang, N. Cao, J .Li, K. Ren and W. Lou, Secure Ranked Keyword Search over Encrypted Cloud Data [C], IEEE International Conference on Distributed Computing Systems, 2010:253-262.

[6] N. Cao, C. Wang, M. Li, et al, Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data [J], Parallel & Distributed Systems IEEE Transactions on, 2011, 25(1):829-837.

[7] W. Sun, B. Wang, N. Cao, et al, Verifiable Privacy-Preserving Multi-Keyword Text Search in the Cloud Supporting Similarity-Based Ranking [J], IEEE Transactions on Parallel & Distributed Systems, 2014, 25(11):71-82.

[8] X. Tian, X. Zhu, P. Shen, et al. An Efficient Ciphertext Search Method Based On Similarity Search Tree [J], Journal of Software, 2016,27(6):1566-1576

[9] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval [J], Information Processing & Management An International Journal, 1988, 24(5):513-523.

[10] G. Salton, A. Wong, C.S. Yang, A vector space model for automatic indexing [M], Morgan Kaufmann Publishers Inc, 1997.

[11] X. Wang, L. Yang, D. Wang and L. Zhen, Improved TF-IDF Keyword Extraction Algorithm [J], Computer Science & application, 2013, 03(1):64-68.

[12] J. Zobel, A.Moffat, Inverted files for text search engines [J]. ACM Computing Surveys, 2010, 38(2):2006.