Journal of Information Hiding and Multimedia Signal Processing ©2018 ISSN 2073-4212
Ubiquitous International Volume **9**, Number **3**, May **2018**

A Multi-Keywords Ranking Searchable Encryption Based on Similarity in Cloud Computing

Yuancheng Li, Boyan Wang, Wenping Chen

School of Control and Computer Engineering
North China Electric Power University
2 Beinong Road, Huilongguan Town, Changping District, Beijing, China
dflyc@163.com, 13261200907@163.com, amo0114@163.com

Received August, 2017; revised February, 2018

Abstract. *In order to ensure the efficient and comprehensive search of cloud-encrypted data, a secure and fast searchable encryption scheme is proposed. The scheme uses a probability public key encryption algorithm to encrypt documents and indexes, in order to complete efficient data encryption without abandoning the data privacy. When calculating the correlation score of keywords and documents, the position weight and the word span are introduced, in order to obtain the better recall rate and accuracy rate. The cloud server obtains the top-k sorted ciphertext collection by calculating the similarity values of the query vector and the searchable index. The simulation results show that the proposed scheme can obtain better search precision, faster encryption and decryption speed and protect user privacy.*
Keywords: cloud computing; Searchable encryption; Data security; Similarity

1. **Introduction.** Cloud computing as a new computing model has the advantages of convenient management, low cost, increasing the storage space without buying any storage devices, therefore more and more companies and users decide to outsource their data to cloud storage. The current successful cloud service platform mainly include Amazon cloud, Microsoft Azure, Alibaba Cloud Computing etc. Due to the cloud server has the right to directly access to users data, more and more users choose to encrypt data firstly and then upload. The attendant problem is how to search encrypted data in the cloud. Downloading the data packets to local to decrypt the file and find the object files, which can cause huge computation and communication overhead, and lower efficiency of search. With the exploration of data security and privacy protection by some domestic and international experts and scholars who are the fields of cryptography and information security in the cloud environment. Searchable encryption technology provides a feasible solution for the search and retrieval of encrypted data.

The concept of searchable encryption is first proposed in Literature [1]. In this scenario, the file contents are encrypted into a series of word matching patterns. The server performs pattern matching operation after user-submitted keywords are encrypted and converted. Obviously, the program requires a linear scan of the encrypted file in this calculation mode so that computation overhead is bigger. A lot of refinement concepts have been put forward since the concept of searchable encryption is proposed, one of them is public key searchable encryption. Literature [2] first proposed an asymmetric searchable encryption scheme based on public key mechanism. The scheme merges the encrypted data indices with public and private key by using the interchangeability of the

exponential in the bilinear operation. Literature [3] proposed a secure file index structure and applied to the field of ciphertext retrieval. The file index in this scenario is encrypted and stored with a Bloom filter. However, the accuracy of the Bloom filter retrieval is not very accurate, it will produce a small probability of misjudgment. Literature [4, 5] proposed a similar security keyword index structure, which establish an encrypted hash index table for the all document. Literature [6] proposed a secure inverted index structure whose basic principle is similar to hash index tables, but the storage methods are different. In our paper, we analyze the inverted index structure and make cloud service provider find target ciphertexts id in the index table according to the query keyword submitted by authorized user. Literature [7] proposed a secure multi-keyword ranking scheme over encrypted files, in which the structure of the index adopt balanced binary tree and can support the dynamic update operation of the document. Literature [8] proposed keyword ranking ciphertext retrieval technology. Literature [9] proposed a keyword-based sorting search scheme. which uses TF×IDF correlation scores to sort the search results and uses one-to-many sequencing mapping technique to protect the keyword word information. Literature [10] adopts matrix linked list to save time due to keys are updated with dynamically updated of keyword dictionaries. Literature [11] uses word frequency information and vector space model to represent the document to support multi-keyword sort retrieval. The scheme build the index structure based on MB tree on the basis of vector space model, which improve the retrieval precision, but it is at the expense of program security. Literature [12] proposed a multi-key ciphertext retrieval scheme, which uses the vector space model to represent document index and query requests, and rank result according to the calculation of the inner value. Literature [13] proposed attribute-based access control scheme with efficient revocation in cloud computing, which two keys are cleverly connected to access policies and subscription strategies. Literature [14] introduced the notion of Public-key Authenticated Encryption with Keyword Search (PAEKS) to solve the problem, in which the data sender not only encrypts a keyword, but also authenticates it. Literature [15] used the strategy that firstly searching the estimated least frequent keyword in the query to significantly narrow down the number of searching documentsthe scheme also supports search results verification. Literature [16] proposed a similarity search method based on search engine a parallel index is established to improve the search efficiency. Literature [17] uses local sensitive hash and searchable symmetric encryption method, it encrypt index high performance using the advanced set based on hash, It not only has the space efficiency but also supports the safely and accurately similarity search. The scheme of literature [18] is to search for similarity of encrypted images. It achieve safe and efficient data retrieval through encrypting images and calculating by secure multi-party. Literature [20] used a keyword weighting algorithm in searchable encryption, which can effectively address keywords that users want to look up, and reduce the time it takes to generate a trapdoor. Literature [21] proposed a frequency hiding query scheme, which only allows the server to view the flattened query distribution, which can protect privacy of data, compared to locality-sensitive hash and searchable symmetric encryption. Literature [19] proposed a Two Round Searchable Encryption (TRSE) scheme that supports ranked multi-keyword search over encrypted data for file retrieval. It employs the vector space model and homomorphic encryption as a result, the information leakage can be eliminated and data security is ensured. However, the computation and communication costs of this scheme are quite large, since every search term in a query requires several homomorphic encryption operations on the data owner side. Further, it uses two-round communication process to retrieve the files back which resulting the unnecessary communication overhead.

In this paper, we study the problem of data security on multi-keywords search in cloud environment. The main work is as follows:

1. We propose an efficient and secure privacy protection scheme, which allows multi-keywords query on encrypted data and returns the relevant sorting documents according to the similarity computation.

2. We use a probabilistic public key encryption algorithm to encrypt documents and indexes, which can better protect data privacy and reduce computation overhead of encryption and decryption.

3. We introduce the location weights and word span into the traditional TF-IDF strategy, and apply it to the preparation of document and the calculation of relevance score between keyword and document.

The rest of the paper is organized as follows: In Section 2, we explain system model and risk model in our scheme. In Section 3, we describe specific search scheme and algorithm. In Sections 4, we analyze experimental results. In Section 5, We summarize our work. The last part of the paper is reference.

2. **Problem Formulation.**

2.1. **Notations definition.** The notations we use in our scheme is shown below.

F – Outsourced file collection $F = \{f_1, f_2, \ldots, f_n\}$, n is the number of files, f_i represents a separate file.

K – Keywords collection $K = \{w_1, w_2, \ldots, w_m\}$, m is the number of keywords.

$\bar{K}$ – The keywords collection which belongs to data user and it is authorized by data owner.

I – Index made up by keywords collection and relevance score.

I'– Encrypted index as well as searchable index.

Ω – The trapdoor generated by searchable keywords.

H – Hash function SHA-1: H: $\{0,1\}^k \times \{0,1\}^* \to \{0,1\}_2^{log(1)}$.

C – Encryption collection of outsource documents $C = \{C_1, C_2, \ldots, C_n\}$.

2.2. **System model.**

In the process of executing the search scheme, One of the most important tasks is to establish a suitable system model, which the process of encrypting, uploading and retrievaling of the outsourced data are based on. The system model consists of three entities: data owner, data user and cloud server. The functions of each entity are as follows:

Data owner: are the user to outsource data to the cloud can be an individual user or an enterprise. They outsource the file collection $F = \{f_1, f_2, \ldots, f_n\}$ in in ciphertext form $C = \{C_1, C_2, \ldots, C_m\}$. They extract key words $K = \{w_1, w_2, \ldots, w_m\}$ from F and the build index I. The encrypted data and index are then uploaded to the cloud.

Authorized user: can retrieve encrypted data in the cloud according to the authorization of the data owner and send the search query Ω to cloud server. Cloud server returns the matched collection and authorized user get the clear text collection $M = \{f_1, f_2, \ldots, f_k\}$.

Cloud Server: is mainly responsible for the storage, search of data owners data and returning the top-k ciphertexts by similarity calculation to users to decrypt it.

2.3. **Risk model.**

In the ciphertext retrieval system, cloud server is supposed to be 'honest but curious'[19], that is the cloud server can provides honest and reliable service, but may be curious about the users data and try to obtain privacy data. This risk enactment is widely used in some existing ciphertext retrieval schemes.

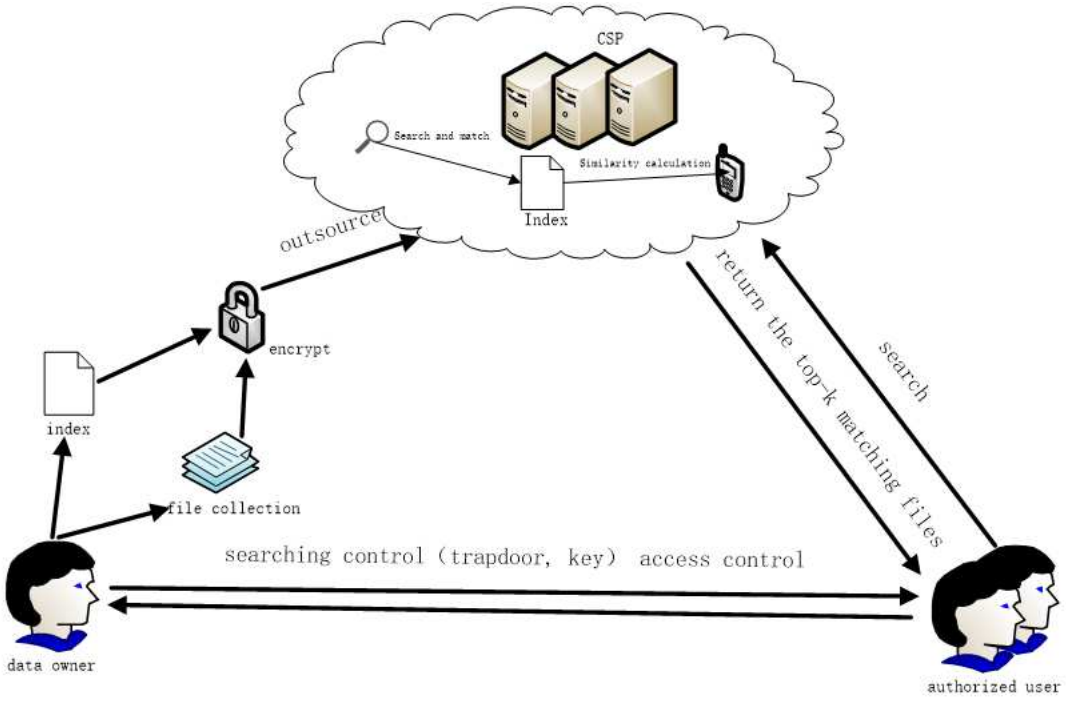

FIGURE 1. System model

Because cloud server may know about some information of the keywords and their TF distribution and etc, and they can deduce keywords index or even file content. We can improve defense measures based on those risks to better protect user privacy.

3. **Multi-keyword Ranking Searchable Encryption Based on Similarity.** Aiming at the data security problem of data search in cloud environment, a searchable encryption scheme(MKSE) based on similarity is proposed. MKSE scheme consists of a preparation phase: key generation, build index, data encryption and a retrieval phase: trapdoor generation, data retrieval, data decryption.

3.1. **Preparation stage.** To avoid some heavy computation overhead, we use the probability public key encryption algorithm[16] to encrypt documents and indexes, which faster than the homomorphic encryption algorithm in TRSE[19] and the fully homomorphic encryption(FHE)[23].We do word segmentation, stop word filtering, low frequency word filtering and keyword weight calculation, and introduce the position weight and the word span into keywords weighting calculation formula, in order to improve the document search rate and accuracy rate. Because the cloud server learns about some keywords and their TF distribution, they can derive keyword indexes and even file contents by using these information. Our solution uses a probability public key encryption algorithm to encrypt the indexes to avoid this problem. The data owner sends the encrypted document to the cloud server along with a searchable index. The preparation phase mainly includes key generation, index construction and data encryption.

$S_1, S_2, r, s, PK \leftarrow keyGen()$: The data owner chooses two random large prime numbers S_1,S_2 to calculate the product of two numbers access to its public key PK, using the extended Euclidean algorithm to obtain its private key r, s.

$I \leftarrow bulidIndex(F, K, r)$: The keyword set K is obtained by scanning the document set F, and record the position m_i and the paragraph number L of the keyword during the scanning process and applied to the document's correlation calculation formula. Adding

the virtual keyword collection Z into the existing keyword collection constitute the keyword dictionary part of index. Put the document id where the keyword is placed into the entry T which go through the pseudo-random replacement. Assigning the relevance value of the keyword to z, and outputs the index I.

$C \leftarrow dataEnc(F, r, PK)$: Convert the document to $f = \{m_1, \ldots, m_n\}$,which m_i is a binary string of length h. Algorithm chooses a random number seed t to get x. and then get the pseudo-random bit p_i. And then its sequence and the document do XOR operation to get ciphertext C.

We use the inverted index in the hash table about the construction of index. The index is made up of keywords table and file list. The keyword table is mainly used to store all keywords, and we add an entry T for each keyword in file list and its value is file id, another part of file list is used to record keywords correlation score in the file. We deduce a label for the j^{th} file with keywords and use the matching j^{th} document to represent a set of K: $M_k = \{k||j : 1 \leq j \leq |F(k)|\}$, where $F(k)$ is the list of matching files. In our design, searching the keyword k is equivalent to searching the label in the form of $k||j$ in M_k.

We use the improved TF-IDF keyword weight calculation method to compute the score of w_i in file f_j . Compare to traditional TF-IDF algorithm, the position weight is added to improve the accuracy of keyword extraction. The calculate formula:

$$\dot{z} = N_i * log(\frac{n}{n_i} + \beta) * m_i \times \frac{l_{wi}}{L} \tag{1}$$

N_i indicates the frequency of keywords w_i in a document f_j ; n represents the total number of documents; n_i represents the number of keywords w_i appears in the document; β represents an empirical value, generally take $0.01, 0.1, 1$; l_{wi} represent the paragraph number where keyword appear in the document; L is the total number of paragraphs; m_i is the location weight of keyword in document. The deployment of keyword in paragraphs mean that the keyword is global or local. The keyword belongs to more paragraphs , the stronger global and the higher score of the keyword has.

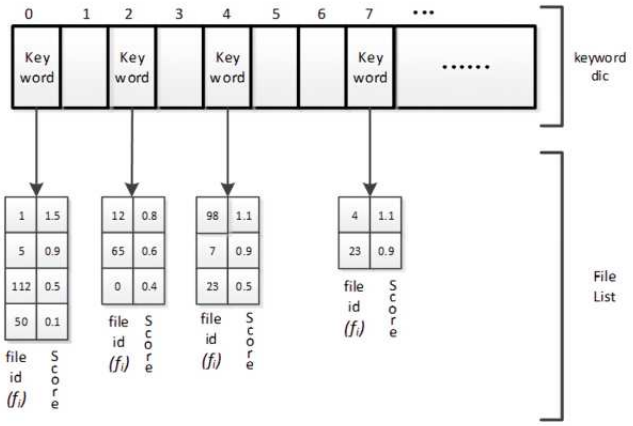

FIGURE 2. Inverted index table

Cloud server may kown about some information of the keywords and their TF distribution and etc, and they can deduce keywords index or even file content. In response to this situation, we decided to add a set of virtual keywords Z to the keywords table to achieve

the effect of increasing the TF-IDF distribution randomness of the original keyword. The index structure is shown as fig-2.

The process of preparation stage:

Preparation stage:	
keyGen():	Select two large random primes S_1,S_2; $PK = S1 \times S2$; Calculate r and s using Euclidian algorithm where $rS_1 + sS_2 = 1$; Output public key is PK, private key is S_1,S_2,r,s.
buildIndex (F, K, r, s):	Scan file collection F; Extract keywords k_1,k_2,...,k_p from F_i; Record keywords location m_i and paragraph number L; For each $k \in K$, build $F(k)$; For each $k_i \in K$: for $1 \leq j \leq \lvert F(k_i)\rvert$: $fid = id(D_{i,j})$, // where $id(D_i, j)$ is the j^{th} identifier in $D(k_i)$ set $T[\pi_r(k_i \lvert\rvert j)] = fid$. // π the pseudo-random permutation For each $D_i \in D$: for each $k_j \in K$: $z_j = TFIDF(k_j)$. Output: $I = \{T, z\}$.

dataEnc (F, r, PK):	Let the file $f = \{m_1, \ldots, m_n\}$ with length n, where each m_i is a binary string of length h and index $I(w_i)$;
	Select t as a seed ; $x_0 = t_2 mod PK$;
	for each i from 1 to n do // Generate the pseudo-random bits
	$x_i = x_{i-1}^2 mod PK$;
	$p_i = x_i mod 2$; // pseudo-random sequence bits
	Calculate $C_i = p_i \oplus m_i$;
	$I'(w_i) = p_i \oplus I(w_i)$.
	End for.
	$x_{n+1} = x_n^2 mod PK$. //generate next random
	output: $C = \{C_1, C_2, \ldots, C_n\}$.

After completing the preparation phase, the data owner uploads the ciphertext set and index $C = \{C_1, C_2, \ldots, C_n, I_{wi}\}$ to the cloud service provider. The calculated bit sequence x_{n+1} is sent to the authorized user or saved in local.

3.2. **Retrieval Stage.** Data user uses the keywords collection $\bar{K}$ authorized by data owner to generate trapdoor by using the Homomorphic encryption and pseudo random permutation. Cloud server matchs the fileid based on searchable query locates index. Cloud server can execute the operation of multi-keywords querying on cloud encrypted data and returning the relevant sorted top-k documents according to the similarity calculation. Data user decrypts the returned ciphertext set.

$\Omega \leftarrow trapdoor(r, K, \bar{K})$: Authorized users use the pseudo-random permutation operation to generate t. At the same time, the user use the key r from the data owner to obtain b with hash function, and combine b and t to generate the query vector Ω.

$C \leftarrow retrieval(\Omega, I)$: The cloud server invokes the algorithm to use the vector value t in the trapoor to find the document set id by positioning the cloud index. For each encrypted document $C_j$cloud server calculates the correlation score vector b in trapdoor

and the value of the TF-IDF value in the index get similarity top-k sorted set $C = \{C_1, C_2, \ldots, C_k\}$. Return it to the user to decrypt.

$M \leftarrow dataDec(S_1, S_2, r, s, PK, C)$: Data user receives the similarity ciphertext set C retrieved from the cloud server, and decrypting to get the plaintext set $M = \{f_1, f_2, \ldots, f_k\}$.

The process of retrieval stage:

Retrieval stage:	
trapdoor $(r, K, \bar{K})$:	For each $k \in \bar{K} \in K$. $t_i = (\pi_r(k_i\|\|1), \ldots, \pi_r(k_i\|\|n), b_i = TFIDF(k_i))$; $b = \sum_{i=1}^{m} H(b_i)^r$; Output: $\Omega = t, b$.
retrieval (Ω, I):	For each i from 1 to $\bar{K}$: $id = T(t)$; $For 1 \le j \le id$: $score = dotprod(z_{c_j}, b)$; From rank$(score_{C_j})$ get top-k C_j; Output: $C = \{C_1, C_2, \ldots, C_k\}$.
dataDec (S_1, S_2, r, s, PK, C):	Compute $d_1 = ((S_1 + 1)/4)^{n+1} mod(S_1 - 1)$; $d_2 = ((S_2 + 1)/4)^{n+1}$; Compute $p = x_{n+1}^{d_1} mod S_1$; $q = x_{n+1}^{d_2} mod S_2$; Compute $x = (prS_1 + qsS_2) mod PK$; For i to n do: $x_i = x_{i-1}^2 mod PK$; p_i is the h least significant bit of x_i; End for. Compute $f_i = p_i \oplus C_i$; Output: $M = \{f_1, f_2, \ldots, f_k\}$.

4. **Performance evaluation.** MRBS scheme use Java language and test in the processor is Intel(R) Core(TM)i5-2430M CPU@2.40GHz, 64 bit windows7 environment. The test document set is Request for comments (RFC) database, which contains more than 6 thousand documents. Our test content is as follows.

4.1. **Preparation stage.** The preparation phase mainly includes the data owner to extract the keyword operation to the document set, the index construction and the data encryption operation. For data encryption, we use the probability of public key encryption algorithm, it is better than the existing scheme FHE which is used to encrypt the documents, because our scheme only needs a modulo can encrypt the h-bit plaintext, which encrypts a larger number of documents with smaller computational overhead and faster processing speed; TRSE scheme requires that the data owner need to do several homomorphic operations for each search term, resulting in a higher cost of computing and communication. Experimental results shown in Figure 3, MKSE encryption program is better than FHE in the processing speed and efficiency.

Fig-4 shows the time consumption of index construction. We need to scan all files to extract keywords set and set up the inverted index table including keywords and file list, we can know index time is proportional to the number of files from figure-4.

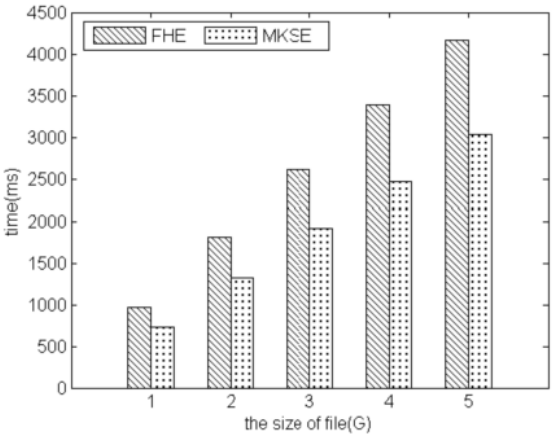

FIGURE 3. the computation cost of encryption files

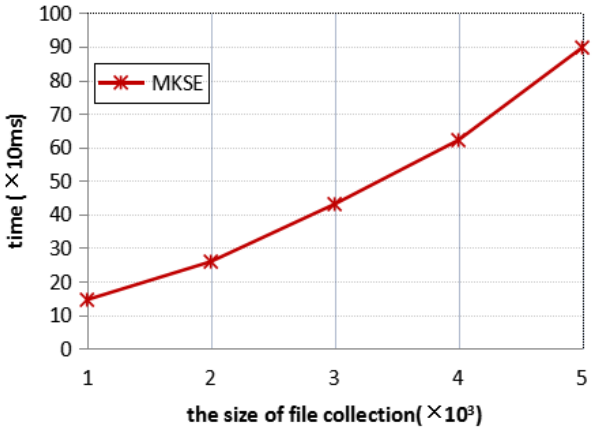

FIGURE 4. the time to build index in different files numbers

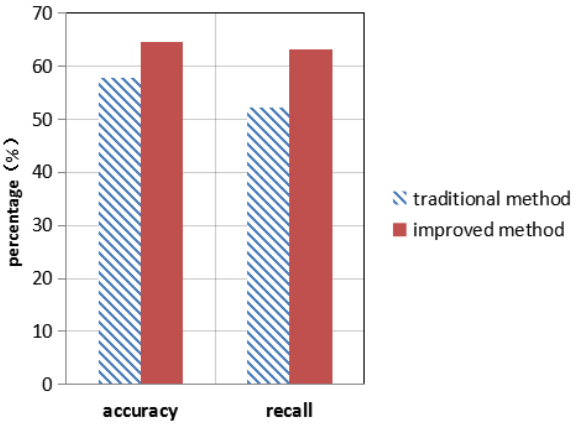

FIGURE 5. the rate of two methods about accuracy, recall, performance

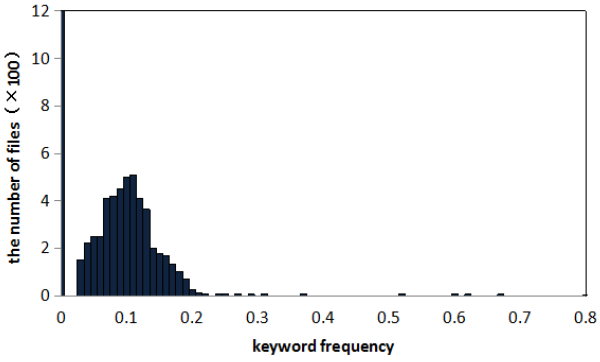

FIGURE 6. the data distribution of the keyword 'network'

Fig-5 is the experimental comparison between traditional TF-IDF method and improved method, the paragraphs and positions of the document are counted at the time of document preprocessing. We can know the improved method has better accuracy, recall and performance from statistical data.

Figure 6 shows the frequency of the selected random keywords such as "network" in the document set, that is each word frequency corresponds to how many documents for specific keywords.

4.2. **Retrieval stage.** Authorized user obtains the weight set t_i of the authorization keyword in each document by the pseudo-random permutation function and with each keyword correlation score bi constitute the query vector Ω. As the TRSE scheme on the authorized set of homomorphic operations led to increased computing,Therefore, the productivity of the MKSE scheme is higher than that of the TRSE scheme. The experimental results are shown in Fig-7.

During the retrieval process, the cloud server only needs to calculate the similarity value of the location document to get a higher correlation scores file collection. TRSE scheme requires homomorphism for each search term, which results in greater computational and

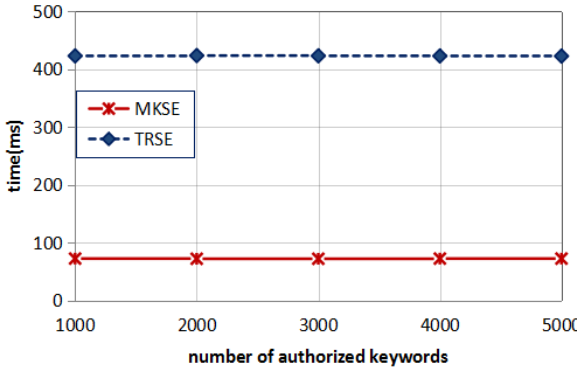

FIGURE 7. the time to generate trapdoor

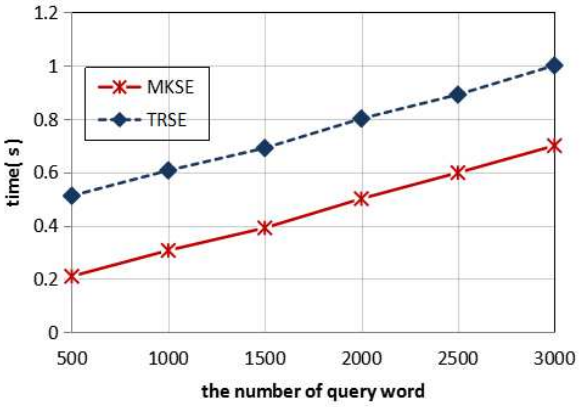

FIGURE 8. the time of searching for files

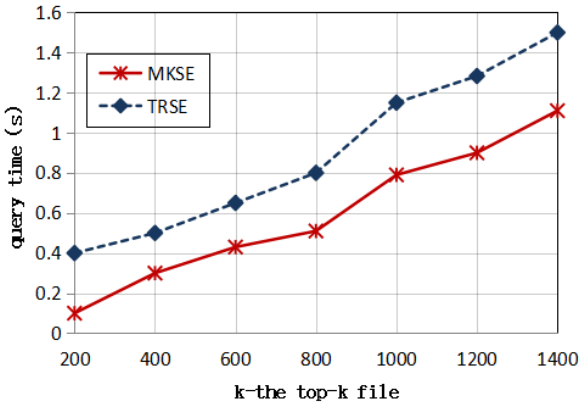

FIGURE 9. the time to return the top-k file

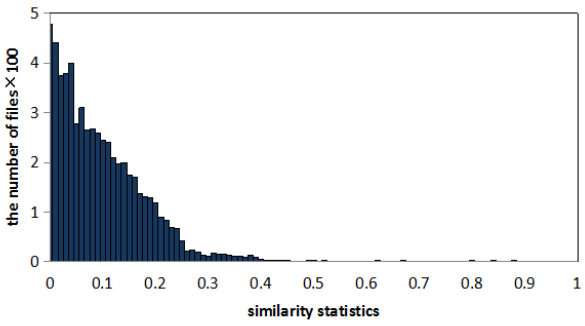

FIGURE 10. similarity statistics of the search keyword 'network'

communication costs. Fig-8 Experimental comparison chart shows that this program can be more quickly retrieve the matching document set.

Figure 9 shows the time consumption of cloud server return top-k document from all matching documents. MKSE schemes similarity calculation method is simpler than TRSE, just perform a point multiplication can get a more matching file set, so the server retrieve top-k document calculation costs is less than the TRSE. Fig-10 shows the similarity statistics of the search keyword 'network'. Fig-11 shows the decryption time consumption of the ciphertext, and we also compare our MKSE with FHE.

The experimental results show that our MRBS scheme is an efficient and secure privacy protection scheme, which allows multi-keywords query on encrypted data and returns the relevant sorting documents according to the similarity computation. The scheme can better protect data privacy, reduce the computational overhead of encryption and decryption and provide authorized user more in line with their query requirements.

5. **Conclusions.** Owing to more and more research at home and abroad, searchable encryption matures. In our scheme, we use the probabilistic encryption algorithm to encrypt the file collection to protect users privacy and reduce the overhead of encryption and decryption. Authorized users use trapdoor to search the matched file. Cloud

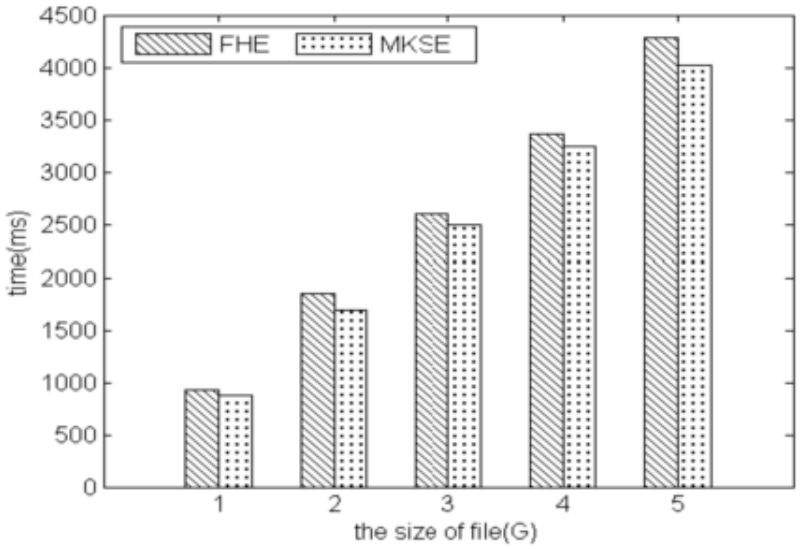

FIGURE 11. the computation cost of decrypting the file

server retrieves top-k files according to similarity calculation for authorized users. The experimental results show that MRBS scheme is an effective way to protect the privacy of data owners, improve the accuracy of the document search and prevent cloud server from obtain additional information.

Acknowledgment. This work is supported by the Fundamental Research Funds for the Central Universities(2018ZD06).

REFERENCES

[1] D. X. Song, D. Wagner, and A. Perrig, Practical Techniques for Searches on Encrypted Data. *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, Berkeley, CA, pp.44–55, 2000.

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, Public Key Encryption with Keyword Search, in *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, pp.506–522, 2004.

[3] E. J. Goh, Secure Indexes, *Submission*, 2004.

[4] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, Searchable symmetric encryption: Improved definitions and efficient constructions, *Journal of Computer Security*, vol.19, no.5, pp.895–934, 2011.

[5] Y. C. Chang, and M. Mitzenmacher, Privacy Preserving Keyword Searches on Remote Encrypted Data, *International Conference on Applied Cryptography and Network Security.*, pp.442–455, 2005.

[6] S. Zerr, and W. Nejdl, Privacy preserving document indexing infrastructure for a distributed environment, vol.1, no.2, p.1638–1643, 2008.

[7] Z. Xia, X. Wang, and Q. Wang, A Secure and Dynamic Multi-Keyword Ranked Search Scheme over Encrypted Cloud Data, *IEEE Transactions on Parallel & Distributed Systems*, vol.27, no.2, pp.340–352, 2016.

[8] A. Swaminathan, Y. Mao, GM Su, H. Gou, A. L. Varna, S. He, M. Wu, and D. W. Oard, *ACM Workshop on Storage Security and Survivability, Storagess 2007, Alexandria, Va, Usa, October*, pp.7–12, 2007.

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

[10] C. Yang, W. Zhang, J. Xu, J. Xu, and N. Yu, A Fast Privacy-Preserving Multi-keyword Search Scheme on Cloud Data, *International Conference on Cloud and Service Computing*, pp.104–110, 2013.

[11] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking, *ACM Sigsac Symposium on Information, Computer and Communications Security*, pp.71–82, 2013.

[12] C. Orencik, M. Kantarcioglu, and E. Savas, A Practical and Secure Multi-keyword Search Method over Encrypted Cloud Data, pp.390–397, 2013.

[13] K. Yang, K. Zhang, X. Jia, M. A. Hasan, and X. Shen, Privacy-preserving attribute-keyword based data publish-subscribe service on cloud platforms, *Information Sciences An International Journal*, vol.387, no.C, pp.116–131, 2016.

[14] Q. Huang, and H. Li, An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks, *Information Sciences*, vol.s 403–404, pp.1–14, 2017.

[15] X. Jiang, J. Yu, J. Yan, and R. Hao, Enabling efficient and verifiable multi-keyword ranked search over encrypted cloud data, *Information Sciences*, vol.s 403–404, pp.22-41, 2017.

[16] Z. Fu, J. Shu, J. Wang , Y. Liu, and S. Lee, Privacy-Preserving Smart Similarity Search Based on Simhash over Encrypted Data in Cloud Computing, *Journal of Internet Technology*, vol.16, no.3, pp.453–460, 2015.

[17] X. Yuan, H. Cui, X. Wang, and C. Wang, Enabling Privacy-Assured Similarity Retrieval over Millions of Encrypted Records, 2015.

[18] H. Liu, and H. Goto, Privacy-Enhanced Similarity Search Scheme for Cloud Image Databases, *Ieice Transactions on Information & Systems*, vol.E99.D, no.12, pp.3188–3191, 2013.

[19] J. Yu, P. Lu, Y. Zhu, G. Xue, and M. Li, Toward Secure Multikeyword Top-k Retrieval over Encrypted Cloud Data, *IEEE Transactions on Dependable & Secure Computing*, vol.10, no.4, pp.239–250, 2013.

[20] Z. Fu, X. Wu, Q. Wang, and K. Ren, Enabling Central Keyword-based Semantic Extension Search over Encrypted Outsourced Data, *IEEE Transactions on Information Forensics & Security*, vol.PP, no.99, pp.1–1, 2017.

[21] X. Yuan, X. Wang, C. Wang, C. Yu, and S. Nutanong, Privacy-Preserving Similarity Joins Over Encrypted Data, *IEEE Transactions on Information Forensics & Security*, vol.12, no.11, pp.2763–2775, 2017

[22] I. H. W. Tten, and T. C. Bell, Managing giga bytes - compressing and indexing documents and images morgan kaufmann publ shers.

[23] M. M. P. Mr, C. A. Dhote, and D. H. S. Mr, Homomorphic Encryption for Security of Cloud Data, *Procedia Computer Science*, vol.79, pp.175–181, 2016.

[24] T. Y. Wu, C. M. Chen, K. H.Wang, J. S. Pan, and W. Zheng, Security Analysis of Rhee et al.'s Public Encryption with Keyword Search Schemes: A Review, *Journal of Network Intelligence*, accepted 2018.

[25] T. Y. Wu, C. Meng, K. H. Wang, C. M. Chen, and J. S. Pan, Comments on Islam Et Al.s Certificateless Designated Server Based Public Key Encryption with Keyword Search Scheme, *International Conference on Genetic and Evolutionary Computing*, pp.199–205, 2017.

[26] T. Y. Wu, C. Meng, C. M. Chen, K. H. Wang, and J. S. Pan, On the Security of a Certificateless Public Key Encryption with Keyword Search, *The Thirteenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP 2017)*, pp.191–197,August 12-15, 2017, Matsue, Shimane, Japan, Springer Smart Innovation, Systems and Technologies, vol.81, pp.191-197, 2018.

[27] T. Y. Wu, F. Meng, C. M. Chen, S. Liu, and J. S. Pan, On the Security of a Certificateless Searchable Public Key Encryption Scheme, *Proc. 10th International Conference on Genetic and Evolutionary Computing (ICGEC 2016)*, pp.113–119, 2016.

[28] T. Y. Wu, T. T. Tsai, and Y. M. Tseng, Efficient searchable ID-based encryption with a designated server, *Annals of telecommunications*, vol.69, no.7-8, pp.391–402, 2014.