INTERNATIONAL JOURNAL OF RESEARCH
In Advanced Engineering Technologies

Volume 6, Issue 1 FEB 2017

ENABLING FINE-GRAINED MULTI-KEYWORD SEARCH SUPPORTING CLASSIFIED SUB-DICTIONARIES OVER ENCRYPTED CLOUD DATA

[1] BODDULA RAMYA,

[2] K. DASARADHA RAMAIAH, M.Tech, (Ph.D), PROFESSOR & HOD, DEPARTMENT OF IT,

[3] NILADRI DEY, M.Tech, ASST.PROFESSOR, DEPT OF IT,

[1] ramya.boddula@gmail.com, [3] niladri.dey@bvrit.ac.in,

[1, 2, 3] B.V.RAJU INSTITUTE OF TECHNOLOGY (AUTONOMOUS)– JNTU HYDERABAD,

Abstract: Using cloud computing, individuals can store their data on remote servers and allow data access to public users through the cloud servers. As the outsourced data are likely to contain sensitive privacy information, they are typically encrypted before uploaded to the cloud. This, however, significantly limits the usability of outsourced data due to the difficulty of searching over the encrypted data. In this paper, we address this issue by developing the fine-grained multi-keyword search schemes over encrypted cloud data. Our original contributions are three-fold. First, we introduce the relevance scores and preference factors upon keywords which enable the precise keyword search and personalized user experience. Second, we develop a practical and very efficient multi-keyword search scheme. The proposed scheme can support complicated logic search the mixed "AND", "OR" and "NO" operations of keywords. Third, we further employ the classified sub-dictionaries technique to achieve better efficiency on index building, trapdoor generating and query. Lastly, we analyze the security of the proposed schemes in terms of confidentiality of documents, privacy protection of index and trapdoor, and unlinkability of trapdoor. Through extensive experiments using the real-world dataset, we validate the performance of the proposed schemes. Both the security analysis and experimental results demonstrate that the proposed schemes can achieve the same security level comparing to the existing ones and better performance in terms of functionality, query complexity and efficiency.

Keywords: Searchable Encryption, Multi-Keyword, Fine-Grained, Cloud Computing.

I. INTRODUCTION

Transmitting the information to the cloud servers the data encryption, though, would considerably lower the usability of data outstanding to the complexity of penetrating over the encrypted data purely encrypting the statistics may still basis other sanctuary concerns. For example, Google Search uses SSL (Secure Sockets Layer) to encrypt the association among search user and Google server when confidential data, such as credentials and emails, appear in the search results. Nevertheless, if the explore user clicks into a different website as of the search consequences page, that website may be talented to categorize the explore terms that the user has worn. Firstly, the statistics owner needs to produce numerous keywords according to the outsourced data. These keywords are then encrypted and stored at the cloud server. When a explore user requirements to admission the outsourced data, it can select some appropriate keywords and send the nothing text of the preferred keywords to the cloud server. The cloud server then uses the cipher text to match the outsourced encrypted keywords, and lastly returns the matching results to the search user. To achieve the similar search efficiency and precision over encrypted data as that of plaintext keyword search, an extensive body of research has been developed in literature. Propose a multi-keyword text search scheme which considers the relevance scores of keywords and utilizes a multidimensional tree technique to achieve efficient search query.

Yu et al. propose a multi-keyword top-k retrieval scheme which uses fully homomorphism encryption to encrypt the index/trapdoor and guarantees high security. Cao et al. propose a multi-keyword ranked search (MRSE), which applies coordinate machine as the keyword matching rule, i.e., return data with the most matching keywords. Although many search functionalities have been developed in previous literature towards precise and efficient searchable encryption, it is still difficult for searchable encryption to achieve the same user experience as that of the plaintext search, like Google search. The relevance scores of keywords can enable more precise returned results, and the preference factors of keywords represent the importance of keywords in the search keyword set specified by search users and correspondingly enables personalized search to cater to specific user preferences. It thus further improves the search functionalities and user experience.

II. EXISTING AND PROPOSED SYSTEMS

A. Existing System

The searchable encryption has been recently developed as a fundamental approach to enable searching over encrypted

cloud data, which precedes the following operations. Wang et al. propose a ranked keyword search scheme which considers the relevance scores of keywords. Sun et al. propose a multi-keyword text search scheme which considers the relevance scores of keywords and utilizes a multidimensional tree technique to achieve efficient search query. Yu et al. propose a multi-keyword top-k retrieval scheme which uses fully homomorphic encryption to encrypt the index/trapdoor and guarantees high security. Cao et al. propose a multi-keyword ranked search (MRSE), which applies coordinate machine as the keyword matching rule, i.e., return data with the most matching keywords.

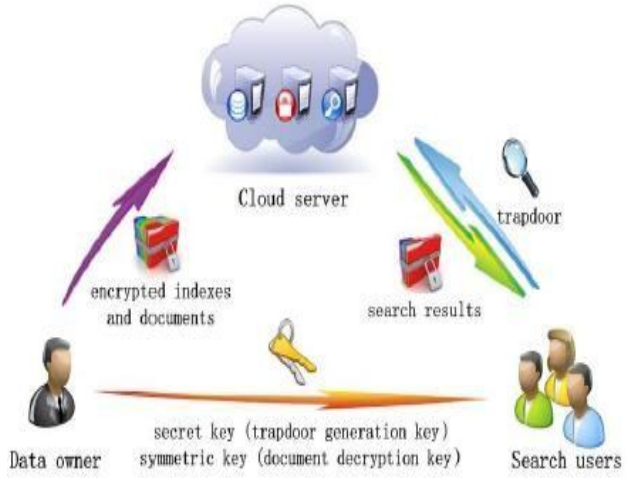

Fig.1. System Architecture

B. Proposed System

In this work, we address by developing two Fine-grained Multi-keyword Search (FMS) schemes over encrypted cloud data. In this system, we introduce the relevance scores and the preference factors of keywords for searchable encryption. The relevance scores of keywords can enable more precise returned results, and the preference factors of keywords represent the importance of keywords in the search keyword set specified by search users and correspondingly enables personalized search to cater to specific user preferences. It thus further improves the search functionalities and user experience. In this system, we realize the "AND", "OR" and "NO" operations in the multi-keyword search for searchable encryption. Compared with schemes, the proposed scheme can achieve more comprehensive functionality and lower query complexity. In this system, we employ the classified sub-dictionaries technique to enhance the efficiency of the above two schemes. Extensive experiments demonstrate that the enhanced schemes can achieve better efficiency in terms of index building, trapdoor generating and query in the comparison with schemes

1. Advantages of Proposed System

- Better search results with multi-keyword query by the cloud server according to some ranking criteria.
- To reduce the communication cost.
- Achieves lower query complexity.
- Achieves better efficiency in index building scheme of our proposed model.

III. MODULE DESCRIPTION

A. Searchable Encryption

This module used on search the key word for encrypted text. This Module Used On Secure Purpose. More Secure for Encrypted Encryption to achieve the same user experience as that of the plaintext search, like Google search. This mainly attributes to following two issues. Firstly, query with user preferences is very popular in the Chipper text search

B. Multi-Keyword

Text search scheme which considers the relevance scores of keywords and utilizes a multidimensional tree technique to achieve efficient search query Multi keyword top-k retrieval scheme which uses fully homomorphic encryption to encrypt the index/trapdoor and guarantees high security Cao et al, propose a multi-keyword ranked search (MRSE), which applies coordinate machine as the keyword matching rule, i.e., return data with the most matching keywords Fine-grained Multi-keyword Search (FMS) schemes over encrypted cloud data. Multi-keyword search and coordinate matching using secure kNN computation scheme multi-keyword top-k retrieval scheme with fully homomorphic encryption, which can return ranked results and achieve high security.

C. Fine-Grained

We propose FMS (CS) schemes which not only support multi-keyword search over encrypted data, but also achieve the fine-grained keyword search with the function to investigate the relevance scores and the preference factors of keywords and, more importantly the logical rule of keywords. In addition, with the classified sub-dictionaries, our proposal is efficient in terms of index building, trapdoor generating and query fine-grained operations of keyword search, i.e., "AND", "OR" and "NO" operations in Google Search, which are definitely practical and significantly enhance the functionalities of encrypted keyword search.

D. Cloud Computing

Cloud computing is a computing term or metaphor that evolved in the late 1900s, based on utility and consumption of computer resources. Cloud computing involves deploying groups of remote servers and software networks that allow different kinds of data sources be uploaded for real time processing to generate computing results without the need to store processed data on the cloud. Clouds can be classified as public, private or hybrid. Synonym expansions are words with the same or similar meanings. In order to improve the accuracy of search results, the A Secure and Dynamic Multi-keyword Ranked extracted from out sourced text documents need to be extended by common synonyms, as cloud customers' searching input might be the synonyms of the predefined A Secure and Dynamic Multi-keyword Ranked, not the exact or fuzzy matching A Secure and Dynamic Multi-keyword Ranked due to the possible synonym substitution and/or her lack of exact knowledge about the data. A common

synonym thesaurus is built on the foundation of the New American Roget's College Thesaurus (NARCT). Then the keyword set is extended by using the constructed synonym thesaurus. Cryptography The art of protecting information by transforming it (encrypting it) into an unreadable format, called cipher text only those who possess a secret key can decipher (or decrypt) the message into plain text. Encrypted messages can sometimes be broken by cryptanalysis, also called code breaking, although modern cryptography techniques are virtually unbreakable.

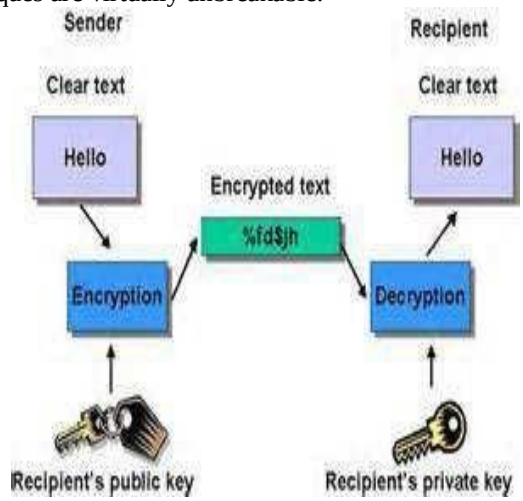

Fig.2. Encryption and Decryption

1. Encryption: In an encryption scheme, the message or information (referred to as *plaintext*) is encrypted using an encryption algorithm, turning it into an unreadable *cipher text* (ibid.). This is usually done with the use of an *encryption key*, which specifies how the message is to be encoded. Any adversary that can see the cipher text, should not be able to determine anything about the original message.

2. Decryption*:* An authorized party, however, is able to decode the cipher text using a decryption algorithm that usually requires a secret decryption key that adversaries do not have access to. For technical reasons, an encryption scheme usually needs a key-generation algorithm, to randomly produce keys. Hence, it is an especially important thing to explore an effective multi-keyword ranked searching service over encrypted outsourced data.

IV. PERFORMANCE EVALUATIONS

In this section, we evaluate the performance of the proposed schemes using simulations, and compare the performance with that of existing proposals in [7]. We apply a real world dataset from the National Science Foundation Research Awards Abstracts 1990-2003, in which we random select multiple documents and conduct real-world experiments on an Intel Core i5 3.2 GHz system.

A. Functionality

We compare functionalities between [7] and our schemes in Table 1, where I and II represent FMS (CS) I and FMS (CS) II, respectively. MRSE [7] can achieve multi-keyword search and coordinate matching using secure kNN computation scheme. And considers the relevance scores of keywords. Compared with the other schemes, our FMS (CS) I consider both the relevance scores and the preference factors of keywords. Note that if the search user sets all relevance scores and preference factors of keywords as the same, the FMS (CS) I degrade to MRSE and the coordinate matching can be achieved. And in the FMS (CS) II, if the search user sets all preference factors of "OR" operation keywords as the same, the FMS (CS) II can also achieve the coordinate matching of "OR" operation keywords particularly, the FMS (CS) II achieves some fine-grained operations of keyword search, i.e., "AND", "OR" and "NO" operations in Google Search, which are definitely practical and significantly enhance the functionalities of encrypted keyword search.

TABLE I. Comparison of Functionalities

	[6]	[13]	[14]	I	II
Multi-keyword search	√	√	√	√	√
Coordinate matching	√	√	√	√	√
Relevance score		√	√	√	
Preference factor				√	√
AND OR NO operations					√

B. Query Complexity

In the FMS (CS) II, we can implement "OR", "AND" and "NO" operations by defining appropriate weights of keywords, this scheme provides a more fine-grained search than [7]. If the keywords to perform "OR", "AND" and "NO" operations are (w'_1, w'_2, · · ·,w'_{l1}), (w''_1 ;w''_2, · · ·,w''_{l2})and (w'''_1, w'''_2,· · · ,w''_{l3}), respectively. Our FMS (CS) II can complete the search with only one query, however, in [7]; they would complete the search through the following steps:

- For the "OR" operation of l_1 keywords, they need only one query *Query* (w'_1, w'_2, · · ·,w'_{l1}) to return a collection of documents with the most matching keywords (i.e., coordinate matching), which can be denoted as X =*Query*(w'_1, w'_2,· · · ,w'_{l1}).
- For the "AND" operation of l_2 keywords, [7] cannot generate a query for multiple keywords to achieve the "AND" operation. Therefore, after costing l_2 queries *Query* (w''_i)(i = 1, 2,· · ·, l_2), they can do the "AND" operation, and the corresponding document set can be denotedas Y=*Query*(w''_1)∩*Query*(w''_2)∩· · ·∩*Query*(w''_{l2}).
- For the "NO" operation of l_3 keywords, they need l_3 queries *Query* (w'''_i) (i = 1, 2, · · ·,l_3), firstly. Then, the document set of the "NO" operation can be denoted as Z = *Query* (w'''_1) ∩ *Query* (w'''_2) ∩· · ·∩ *Query* (w'''_{l3}).
- Finally, the document collection achieved "OR", "AND" and "NO" operations can be represented as X ∩Y ∩Z.

As shown in Fig. 3a, 3b and 3c, to achieve these operations, the FMS (CS) II can outperform the existing proposals with less queries generated.

C. Efficiency

1. Computation Overhead

In order to easily demonstrate our scheme computation overhead, we analysis our scheme from each phase index

building note that the *Index building* phase of [7] is the same as our FMS II scheme, without calculating the relevance score. And the *Index building* phase of the FMS I is the same as, containing the relevance score computing. Compared with the FMS I, the FMS II do not need to calculate the relevance score. And compared with the computation cost of building index, the cost of calculating the relevance score is negligible, we do not distinguish them. Moreover, in our enhanced schemes (FMSCS), we divide the total dictionary into 1 common sub-dictionary and 20 professional sub-dictionaries (assume each data owner averagely chooses 1 common sub-dictionary and 3 professional sub-dictionaries to generate the index). As shown in Fig. 4, we can see the time for building index is dominated by both the size of dictionary and the number of documents. And compared with [7], and our FMS schemes, the FMSCS schemes largely reduce the computation overhead. Trapdoor generating: In Trapdoor generating phase, [7] firstly creates a vector according to the search keyword set$W$, then encrypts the vector by the secure kNN computation scheme. And also generates a vector and uses homomorphic encryption to encrypt each dimension.

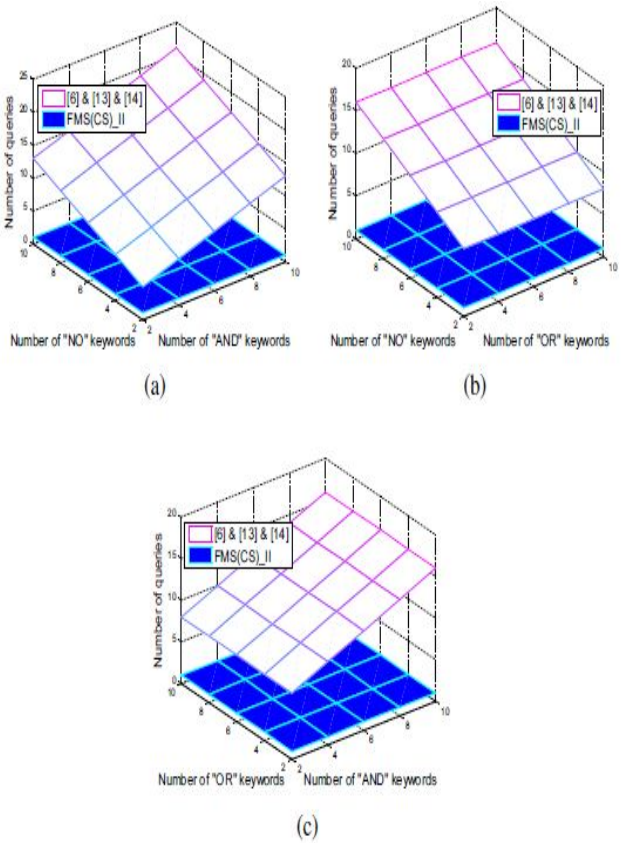

Fig.3. Time for Building Index. (a) Number of Queries for the Different Number of "AND" and "NO" Keywords with the Same Number of "OR" Keywords, L_1 = 5. (b) Number of Queries for the Different Number of "OR" and "NO" Keywords with the Same Number of "AND" Keywords, L_2 = 5. (C) Number of Queries for the Different Number of "AND" and "OR" Keywords with the Same Number of "NO" Keywords, L_3 = 5.

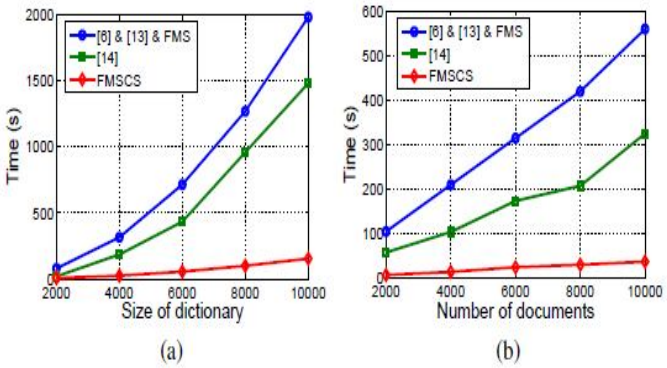

Fig.4. Time for Building Index. (a) For the Different Size of Dictionary with the Same Number of Documents, N=6000. (b) For the Different Number of Documents with the Same Size of Dictionary, $|W|$ = 4000.

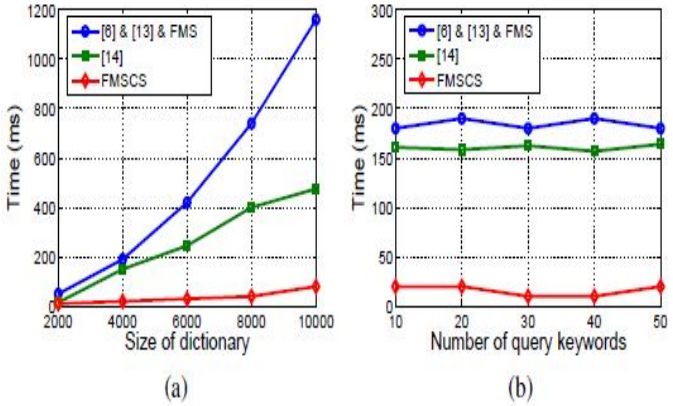

Fig.5. Time for Generating Trapdoor. (a) For the Different Size of Dictionary with the Same Number of Query Keywords, $|W|$=20. (b) For the Different Number of Query Keywords with the Same Size of Dictionary, $|W|$ = 4000.

In comparison, our FMS I and FMS II schemes should firstly generate a super-increasing sequence and a weight sequence, respectively. But actually, we can pre-select a corresponding sequence for each scheme, it can also achieve search process and privacy. Because even if the vectors are the same for multiple queries, the trapdoors will be not the same due to the security of kNN computation scheme. Therefore, the computation cost of [7] and all FMS schemes in *Trapdoor generating* phase are the same. As shown in Fig. 5, the time for generating trapdoor is dominated by the size of dictionary, instead of the number of query keywords. Hence, our FMSCS schemes are also very efficient in *Trapdoor generating* phase. Query. As [7] the FMS all adopt the secure kNN computation scheme, the time for query is the same. The computation overhead in *Query* phase, as shown in Fig. 6, is greatly affected by the size of dictionary and the number of documents, and almost has no relation to the number of query keywords. Further we can see, our FMSCS schemes significantly reduce the computation cost in *Query* phase. As needs to encrypt each dimension of index/trapdoor using full homomorphic encryption, its index/trapdoor size is enormous. Note that, in *Trapdoor generating* and *Query* phases, the

computation overheads are not affected by the number of query keywords. Thus our FMS and FMSCS schemes are more efficient compared with some multiple keyword search schemes, as their cost is linear with the number of query keywords.

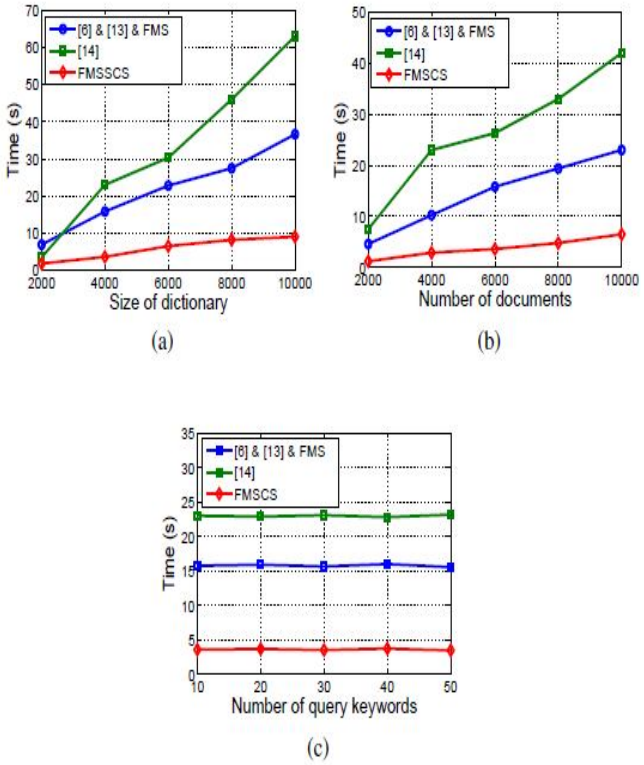

Fig.6. Time for Query. (a) For the Different Size of Dictionary with the Same Number of Documents and Number of Search Keywords, N = 6000; $|W|$ = 20. (b) for the Different Number of Documents with the Same Size of Dictionary and Number of Search Keywords, $|W|$ = 4000; $|W|$ = 20. (c) For the Different Number of Search Keyword with the Same Size of Dictionary and Number of Documents, N = 6000; $|W|$ = 4000.

2. Storage Overhead

As shown in Table 2, we provide a comparison of storage overhead among several schemes. Specifically, we evaluate the storage overhead from three parts: the data owner, the search user and the cloud server. According to Table 2, in the FMS, the FMSCS as well as schemes of [7], the storage overhead of the data owner is the same. In these schemes, the data owner preserves her secret key $K = (S, M_1, M_2)$ and symmetric key sk locally, where S is an $(m+1)$-dimensional vector, M_1 and M_2 are $(m+1) \times (m+1)$ invertible matrices. All elements in S, M_1 and M_2 are the float number. Since the size of a float number is 4 bytes, the size of K is $4 \cdot (m+1) + 8 \cdot (m + 1)2$ bytes. We assume that the size of sk is S_{sk} that is a constant. Thus, the total size of storage overhead is $4 \cdot (m+1) + 8 \cdot (m + 1)2 + S_{sk}$ bytes. However, in [14], the storage overhead of data owner is $\lambda^5/8$ bytes, where the λ is the secure parameter. The storage overhead is 4GB when we choose λ = 128, which is popular in a full homomorphic encryption scheme. However, the storage overhead of the FMS and the FMSCS are almost 763MB when we choose m = 10000, which is large enough for a search scheme. Therefore, the FMS and the FMSCS are more efficient than scheme in terms of the storage overhead of the data owner. As shown in Table II, a search user in the FMS, the FMSCS as well as the schemes of [7] preserves the secret key $K = (S, M_1, M_2)$ and the symmetric key sk locally.

Therefore, the total storage overhead is $4(m+1)+8(m + 1)^2+S_{sk}$ bytes. However, in, the storage overhead is $\lambda^5/8+ \lambda^2/8$ bytes. The storage overhead is 4GB when we choose λ = 128, which is popular in a full homomorphic encryption scheme. However, the storage overhead of the FMS and the FMSCS are almost 763MB when we choose m = 10000, which is large enough for a search scheme. Therefore, the FMS and the FMSCS are more efficient than scheme in terms of the storage overhead of the search user. The cloud server preserves the encrypted documents and the indexes. The size of encrypted documents in all schemes are the same, i.e., $N \cdot Ds$. For the indexes, in the FMS and schemes in [7], the storage overhead are $8 \cdot (m+1) \cdot N$ bytes. In the FMSCS, the storage overhead is $8 \cdot \varepsilon \cdot (m+1) \cdot N$ bytes, where $0 < \varepsilon < 1$. When m = 1000 and N = 10000 which are large enough for a search scheme, the storage overhead of indexes is about 132MB in the FMSCS. And in schemes of [7] as well as the FMS, the size of indexes is 760MB with the same conditions. In scheme the storage overhead of indexes is $N \cdot D_s + m \cdot N \cdot (\lambda/8)^5$ bytes, it is 4GB when we choose λ = 128, which is popular in a full homomorphic encryption scheme. Therefore, the FMS and the FMSCS are more efficient than scheme in terms of the storage overhead of the cloud server.

3. Communication Overhead

As shown in Table 3, we provide a comparison of communication overhead among several schemes. Specifically, we consider the communication overhead from three parts: the communication between the data owner and the cloud server (abbreviated as D-C), the communication between the search user and the cloud server (abbreviated as C-S) and the communication between the data owner and the search user (abbreviated as D-S). D-C. In the FMS as well as schemes of [7], the data owner needs to send information to cloud server in the form of $C_j \,||FID_j\,||I_j$ (j = 1; 2; $\cdots$;N), where the C_j represents the encrypted documents, FID_j represents the identity of the document and I_j represents the index. We assume that the average size of documents is D_s, thus the size of documents is $N \cdot D_s$. We assume the encrypted documents identity FID is a 10-byte string. Thus, the total size of the identity FID is $10 \cdot N$ bytes. The index $I_j = (paM_1, pbM_2)$ contains two (m+1)-dimensional vectors. Each dimension is a float number (the size of each float is 4 bytes). Thus, the total size of index is $8 \cdot (m+1) \cdot N$ bytes. Therefore, the total size of communication overhead is $8 \cdot (m+1) \cdot N+10 \cdot N+N \cdot D_s$ bytes. In the FMSCS, the total size of communication overhead is $8 \cdot \varepsilon \cdot (m+1) \cdot N +10 \cdot N+N \cdot D_s$ bytes. If we choose the ε as 0.2, the size of index is $1.6 \cdot (m+1) \cdot N$ bytes, and the total size of communication of FMSCS is $1.6 \cdot$

$(m+1)\cdot N+10\cdot N+D_s\cdot N$ bytes. However, in, the communication overhead is $N\cdot Ds+m\cdot N\cdot \lambda^5/8$ bytes, where λ is the secure parameter. If we choose $\lambda = 128$ which is popular in a full homomorphic encryption scheme and $m = 1000$ and $N = 10000$ which are large enough for a search scheme, the FMS and the FMSCS are more efficient than scheme in terms of the communication overhead of D-C.

TABLE II. Comparison of Storage Overhead (Bytes). (*M* Represents the Size of Dictionary; *N* Represents the Number of Documents; D_s Represents the Average Size of Each Encrypted Document; Λ Represents the Secure Parameter; " Represents the Decrease Rate of Dictionary By Using Our Classified Sub-Dictionaries Technology; S_{sk} Represents the Size of Symmetric Key.)

	[14]	[6], [13] and FMS	FMSCS
Data Owner	$\lambda^5/8$	$4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$	$4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$
Search User	$\lambda^5/8+\lambda^2/8$	$4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$	$4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$
Cloud Server	$N\cdot D_s+m\cdot N\cdot\lambda^5/8$	$N\cdot D_s+8\cdot(m+1)\cdot N$	$N\cdot D_s+8\cdot\varepsilon\cdot(m+1)\cdot N$

TABLE III. Comparison of Communication Overhead (Bytes). (*M* Represents the Size Of Dictionary; *N* Represents the Number of Documents; D_s Represents the Average Size of Each Encrypted Document; *T* Represents the Number of Returned Documents; Λ Represents the Secure Parameter; " Represents the Decrease Rate of Dictionary by using our Classified Sub-Dictionaries Technology; S_{sk} Represents the Size of Symmetric Key.)

	[14]	[6], [13] and FMS	FMSCS
D-C	$N\cdot D_s+m\cdot N\cdot\lambda^5/8$	$8\cdot(m+1)\cdot N+10\cdot N+N\cdot D_s$	$8\cdot\varepsilon\cdot(m+1)\cdot N+10\cdot N+N\cdot D_s$
C-S	$m\cdot\lambda^5/8+T\cdot D_s$	$8\cdot(m+1)+T\cdot D_s$	$8\cdot\varepsilon\cdot(m+1)+T\cdot D_s$
D-S	$\lambda^5/8+\lambda^2/8$	$4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$	$4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$

C-S. The C-S consists of two phases: *Query* and *Results returning*. In the *Query* phase, a search user in the FMS as well as the schemes in [7] sends the trapdoor to the cloud server in the form of $T_{\sim W}=(M_1^{-1}q_a, M_2^{-1}q_b)$, which contains two (m+1)-dimensional vectors. Thus, the communication overhead is $8\cdot(m+1)$ bytes. In the FMSCS, the communication overhead is $8\cdot\varepsilon\cdot(m+1)(0<\varepsilon<1)$ bytes. In the *Results returning* phase, the cloud server sends the corresponding result to the search user. The communication overhead of CS increases along with the number of returned documents at this point. We assume that the number of the returned documents is *T*, thus, the total communication overhead of cloud server to search user is $T\cdot Ds$ bytes. Therefore, the total communication overhead of C-S is $8\cdot m+T\cdot Ds$ bytes. In the FMS as well as the schemes in [7], the total communication overhead of C-S is $8\cdot\varepsilon\cdot(m+1)+T\cdot D_s$ bytes. In, the total communication overhead of C-S is $m\cdot\lambda^5/8+T\cdot Ds$ bytes. If we choose $\lambda = 128$ which is popular in a full homomorphic encryption scheme and $m = 1000$ and $N = 10000$ which are large enough for a search scheme, the FMS and the FMSCS are more efficient than scheme in terms of the communication overhead of C-S. D-S. From table 3, we can see that the communication overhead of the FMS, the FMSCS as well as schemes in [7] is the same. In the *Initialization* phase, the data owner sends the secret key $K = (S, M_1, M_2)$ and symmetric key *sk* to the search user, where *S* is an (*m*+ 1)- dimensional vector, M_1 and M_2 are $(m + 1) \times (m + 1)$ invertible matrices. Thus, the size of the secret key *K* is $4\cdot(m+1)+8\cdot(m+1)^2$ bytes. Therefore, the total size of communication overhead is $4\cdot(m+1)+8\cdot(m+1)^2+S_{sk}$ bytes, where the S_{sk} represents the size of symmetric key. However, the communication overhead of scheme is $\lambda^5/8+\lambda^2/8$ bytes. The communication overhead is 4GB when we choose $\lambda = 128$, which is popular in a full homomorphic encryption scheme. However, the communication overhead of the FMS and the FMSCS are almost 763MB when we choose $m = 10000$, which is large enough for a search scheme. Therefore, the FMS and the FMSCS are more efficient than scheme in terms of the communication overhead of D-S.

V. CONCLUSION

We have examined on the fine-grained multi keyword search (FMS) subject over encrypted cloud data, and future two FMS schemes. The FMS I includes both the significance scores and the partiality factors of keywords to augment more accurate search and enhanced users' experience, in that order. The FMS II realize secure and competent search with realistic functionality, i.e., "AND", "OR" and "NO" operations of keywords. In addition, we have planned the better schemes behind confidential sub-dictionaries (FMSCS) to advance competence. For the future work, we propose to add expand the application to reflect on the extensibility of the file set and the multi-user cloud environments. Towards this trend, we have made some beginning consequences on the extensibility and the multiuser cloud environments. Another remarkable topic is to increase the greatly scalable searchable encryption to enable able explore on large realistic databases.

VI. REFERENCES

[1]Hongwei Li, Member, IEEE, Yi Yang, Student Member, IEEE, Tom H. Luan, Member, IEEE, Xiaohui Liang, Student Member, IEEE, Liang Zhou, Member, IEEE, and Xuemin (Sherman) Shen, Fellow, IEEE, "Enabling Fine-grained Multi-keyword Search Supporting Classified Sub-dictionaries over Encrypted Cloud Data", IEEE Transactions on Dependable and Secure Computing, 2015.

[2]H. Liang, L. X. Cai, D. Huang, X. Shen, and D. Peng, "An SMDP based service model for inter domain resource allocation in mobile cloud networks," IEEE Transactions on Vehicular Technology, vol. 61, no. 5, pp. 2222–2232, 2012.

[3]M. M. Mahmoud and X. Shen, "A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 10, pp. 1805–1818, 2012.

[4]Q. Shen, X. Liang, X. Shen, X. Lin, and H. Luo, "Exploiting redistributed clouds for e-health monitoring system with minimum service delay and privacy preservation," IEEE Journal of Biomedical and Health Informatics, vol. 18, no. 2, pp. 430–439, 2014.

[5]T. Jung, X. Mao, X. Li, S.-J. Tang, W. Gong, and L. Zhang, "Privacy preserving data aggregation without secure

channel: multivariate polynomial evaluation," in Proceedings of INFOCOM. IEEE, 2013, pp. 2634–2642.
[6]Y. Yang, H. Li, W. Liu, H. Yang, and M. Wen, "Secure dynamic searchable symmetric encryption with constant document update cost," in Proceedings of GLOBCOM. IEEE, 2014, to appear.
[7]N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 1, pp. 222–233, 2014.
[8]https://support.google.com/websearch/answer/173733?hl=en.
[9]D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proceedings of S&P. IEEE, 2000, pp. 44–55.
[10]R. Li, Z. Xu, W. Kang, K. C. Yow, and C.-Z. Xu, "Efficient multi-keyword ranked query over encrypted data in cloud computing," Future Generation Computer Systems, vol. 30, pp. 179–190, 2014.
[11]H. Li, D. Liu, Y. Dai, T. H. Luan, and X. Shen, "Enabling efficient multi-keyword ranked search over encrypted cloud data through blind storage," IEEE Transactions on Emerging Topics in Computing, 2014, DOI10.1109/ TETC.2014 .2371239.
[12]C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in Proceedings of ICDCS. IEEE, 2010, pp. 253–262.