Advanced Science and Technology Letters
Vol.31 (MulGraB 2013), pp.284-289
http://dx.doi.org/10.14257/astl.2013.31.58

# An Efficient and Privacy-Preserving Semantic Multi-Keyword Ranked Search over Encrypted Cloud Data

Zhihua Xia, Li Chen, Xingming Sun, and Jin Wang

xia_zhihua@163.com, z_chenli@163.com, sunnudt@163.com and wangjin@nuist.edu.cn
Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science & Technology, Nanjing, 210044, China
School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing, 210044, China

**Abstract.** As so much advantage of cloud computing, more and more data owners centralize their sensitive data into the cloud. In this paper, we propose a semantic multi-keyword ranked search scheme over the encrypted cloud data, which simultaneously meets a set of strict privacy requirements. Firstly, we utilize the “Latent Semantic Analysis” to reveal relationship between terms and documents. The relationship between terms is automatically captured. Secondly, our scheme employ secure “k-nearest neighbor (k-NN)” to achieve secure search functionality. The proposed scheme could return not only the exact matching files, but also the files including the terms latent semantically associated to the query keyword. Finally, the experimental result demonstrates that our method is better than the original MRSE scheme.

## 1 Introduction

Due to the rapid expansion of data, the data owners tend to store their data into the cloud to release the burden of data storage and maintenance[1]. However, as the cloud customers and the cloud server are not in the same trusted domain, our outsourced data may be under the exposure to the risk. Thus, before sent to the cloud, the sensitive data needs to be encrypted to protect for data privacy and combat unsolicited accesses.

Fuzzy keyword searches [2-4] have been developed. Chuah et al. [2] propose a privacy-aware bed-tree method to support fuzzy multi-keyword search. This approach uses edit distance to build fuzzy keyword sets. Bloom filters are constructed for every keyword. Then, it constructs the index tree for all files where each leaf node a hash value of a keyword. Li et al. [3] exploit edit distance to quantify keywords similarity and construct storage-efficient fuzzy keyword sets. Specially, the wildcard-based fuzzy set construction approach is designed to save storage overhead. Wang et al. [4] employ wildcard-based fuzzy set to build a private trie-traverse searching index. These fuzzy search methods support tolerance of minor typos and format inconsistencies, but do not support semantic fuzzy search. Considering the existence of polysemy and synonymy [5] , the model that supports multi-keyword ranked search and semantic search is more reasonable.

In this paper, we will solve the problem of multi-keyword latent semantic ranked search over encrypted cloud data and retrieve the most relevant files. We define a new scheme named Latent Semantic Analysis (LSA)-based multi-keyword ranked search which supports multi-keyword latent semantic ranked search. By using LSA, the proposed scheme could return not only the exact matching files, but also the files including the terms latent semantically associated to the query keyword.

The reminder of this paper is organized as follows. In section 2, we describe the system model, privacy requirements, and notations. Section 3 provides the detailed description of our proposed mechanism. Section 4 presents the experiment and security analysis. Section 5 summarizes the conclusion.

## 2 Problem Formulation

### 2.1 System Model

The system model can be considered as three entities, as depicted in Fig .1: the data owner, the data user and the cloud server. Data owner has a collection of data documents $D = \{d_1,d_2,...,d_m\}$ .A set of distinct keywords $W == \{w_1,w_2,...,w_n\}$ is extracted from the data collection $D$ . The data owner will firstly construct an encrypted searchable index *I* from the data collection $D$ .Then, the data owner upload both the encrypted index *I* and the encrypted data collection *C* to the cloud server. Data user provides *t* keywords for the cloud server. The cloud server only sends back top-*l* files that are most relevant to the search query.

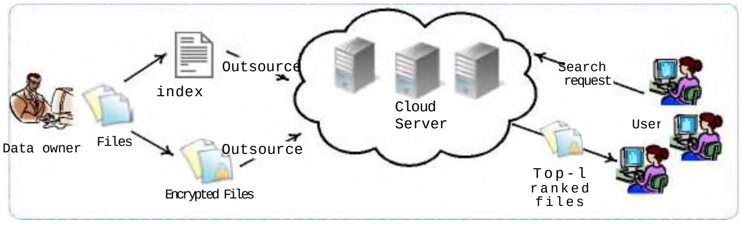


**Fig. 1.** Architecture of ranked search over encrypted cloud data

### 2.2 Threat models and Design Goals

The cloud server both follows the designated protocol specification but at the same time analyzes data in its storage and message flows received during the protocol so as to learn additional information.

The designed goals of our system are following:

**Latent Semantic Search:** We use statistical techniques to estimate the latent semantic structure, and get rid of the obscuring “noise” [5].

**Multi-keyword Ranked Search:** It supports both multi-keyword query and support result ranking.

**Privacy-Preserving:** Our scheme is designed to meet the privacy requirement and prevent the cloud server from learning additional information from index and trapdoor.

*1)Index Confidentiality*. The *TF* values of keywords are stored in the index. Thus, the index stored in the cloud server needs to be encrypted;

*2)Trapdoor Unlinkability*. The cloud server should not be able to deduce relationship between trapdoors.

*3)Keyword Privacy*. The cloud server could not discern the keyword in query, index by analyzing the statistical information like term frequency.

## 2.3 Notations and Preliminaries

- *D* --the plaintext document collection, denoted as a set of n data documents D = $\{d_1, d_2, ..., d_m\}$
- *C* --the encrypted document collection stored in the cloud server, denoted as C = $\{c_1, c_2, ..., c_m\}$ .
- W=--the dictionary, the keyword set composing of m keyword, denoted as W== $\{w_1, w_2, ..., w_n\}$ .
- *I* --the searchable index associated $C$, denoted as $(I_1, I_2, ..., I_m)$ .
- $tf_{i,j}$--the term frequency, the ***i-th*** term appears times in the ***j-th*** document.
- $\mathbf{A}'[j]$ --the data vector for document $d_j$ where the element $\mathbf{A}'[i][j]$ represents the term frequency $tf_{i,j}$ of the corresponding keyword $i$W=in document $d_j$ .
- **Q** --the query vector indicating the keywords of interest where each bit $\mathbf{Q}[j] \in \{0,1\}$ represents the existence of the corresponding keyword in the query $\tilde{W}$ .

***Latent Semantic Analysis*:** In information retrieval, latent semantic analysis is a solution for discovering the latent semantic relationship. It adopts singular-value decomposition, which is abbreviated as *SVD* to find the semantic structure between terms and documents. In this paper, the term-document matrix consists of *n* rows, each of which represents the data vector for each file,

$$\mathbf{A}' = (\mathbf{A}'[1] \ \dots \ \mathbf{A}'[j] \ \dots \ \mathbf{A}'[m]) \tag{1}$$

as depicted in the Eq.1. Then, we take a large term-document matrix and decompose it into a set of $k$ , orthogonal factors from which the original matrix can be approximated by linear combination [5]. For example, a term-document matrix named $\mathbf{A}'$ can be decomposed into the product of three other matrices:

$$\underset{n\times m}{\mathbf{A}'} = \underset{n\times t}{\mathbf{U}'} \cdot \underset{t\times t}{\mathbf{S}'} \cdot \underset{t\times m}{\mathbf{V}'} \tag{2}$$

such that $\mathbf{U}'$ and $\mathbf{V}'$ have orthonormal columns, $\mathbf{S}'$ is diagonal. We choose previous $k$ columns of $\mathbf{S}'$ , and then deleting the corresponding columns of $\mathbf{U}'$ and $\mathbf{V}'$ respectively.The result is a reduced model:

$$\mathbf{A}_{n\times m} = \mathbf{U}'_{n\times k}\mathbf{S}'_{k\times k}\mathbf{V}'_{k\times m} \approx \mathbf{A}' \quad (3)$$

which is the rank- $k$ model with the best possible least-squares-fit to $\mathbf{A}'$ [5].

***Secure k-NN***: In order to compute the inner product in a privacy-preserving method, we will adapt the secure $k$ -nearest neighbor scheme[6]. This splitting technique is secure against known-plaintext attack, which is roughly equal in security to a $d$-bit symmetric key. We will get the details from [6, 7].

## 3 Proposed Scheme

### 3.1 Our Scheme

According to the above definition about *LSA*, the data owner builds a term-document matrix $\mathbf{A}'$ . We reduce the dimensions of the original matrix $\mathbf{A}'$ to get a new matrix $\mathbf{A}$ which is calculated the best “reduced-dimension” approximation to the original term-document matrix[5]. Specially, $\mathbf{A}[j]$ denotes the $j$ *-th* column of the matrix $\mathbf{A}$ .

-  **Setup** The data owner generates a $n$ +2 -bit vector as **X** and two $(n + 2)\times(n + 2)$ invertible matrices $\{\mathbf{M}_1, \mathbf{M}_2\}$ .The secret key *SK* is the form of a 3-tuple as $\{\mathbf{X}, \mathbf{M}_1, \mathbf{M}_2\}$.
-  **BuildIndex(A′,** ***SK*****)** The data owner extracts a term-document matrix $\mathbf{A}'$. Following, we multiply these three matrices to get the result matrix $\mathbf{A}$ .Taking privacy into consideration, it is necessary that the matrix $\mathbf{A}$ is encrypted before outsourcing. After applying dimension-extending, the original $\mathbf{A}[j]$ is extended to $(n + 2)$ -dimensions, instead of $n$ . Namely, the $(n + 1)$ *-th* entry in $\mathbf{A}[j]$ is set to a random number $e_j$ , and the $(n + 2)$ *-th* entry in $\mathbf{A}[j]$ is set to 1 during the dimension extending. Finally, $\mathbf{A}[j]$ can be represented as $((\mathbf{A}[j])^T, e_j, 1)^T$ . The subindex $I_j = \{\mathbf{M}_1^T \mathbf{A}'[j], \mathbf{M}_2^T \mathbf{A}''[j]\}$ is built.
-  **Trapdoor($\tilde{W}$)** With $t$ keywords of interest in $\tilde{W}$ as input, one binary vector **Q** is generated. The $(n + 1)$ *-th* entry in **Q** is set to a random number 1, and then scaled by a random number $r \neq 0$, and the $(n+2)$ *-th* entry in **Q** is set to another random number $t$. **Q** can be represented as $(r\mathbf{Q}^T, r, t)^T$. The trapdoor $T_{\tilde{w}}$ is generated as $\{\mathbf{M}_1^{-1}\mathbf{Q}', \mathbf{M}_2^{-1}\mathbf{Q}''\}$ .
-  **Query($T_{\tilde{w}}$,*l*,*I*)** The inner product of $I_j$ and $T_{\tilde{w}}$ is calculated by the cloud server. After sorting all scores, the cloud server returns the top-*l* ranked id list to the data

user. The final similarity scores would be:

$$I \oplus T_{j w^{\sim}} = \quad \mathbf{ATM}\ \{\mathbf{m}_1^{\square 1}, \mathbf{M}_2^{\square 1}\ \mathbf{Q} \oplus 2\}$$

$$(\mathbf{A}[\ ])2^T\ j\ \ \mathbf{Q}\ \ \mathbf{A}^T\ \ 22\oplus \quad 2 \oplus \mathbf{2} + (\ [\ j\ ])\ \mathbf{Q} \qquad (4)$$

$$(\ [\ ])^T\ (\mathbf{A}\ [], j, 1)\ (\mathbf{Q}\mathbf{Q}\oplus \quad )\quad j \textstyle\sum_j \quad r\ ;^T r\ ,\ t \oplus^T$$

$$= r(\mathbf{A}[j\ ]\ \mathbf{Q}^T \oplus \textstyle\sum_j + )\ t\ +$$

## 4 Performance Analysis

In this section, we show a thorough experimental evaluation of the proposed technique on a real dataset: the *MED* dataset.

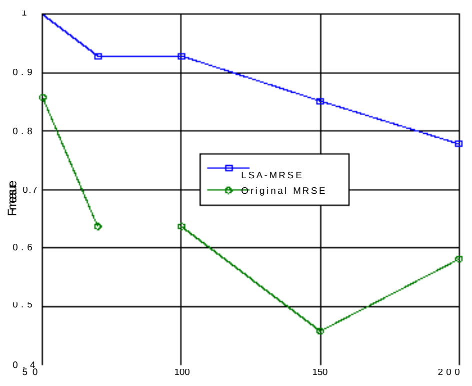
### 4.1 Performance Analysis

$$F = \frac{\text{F-measure that combines precision and recall is the harmonic mean of}}{} \qquad (5)$$

precision and recall[8]. Here, we adopt F-measure to weigh the result of our experiments.

*2⊕ precision⊕ recall*

*precision+recall*



For a clear comparison, our proposed scheme attains score higher than the original *MRSE* in F-measure. Since the original scheme employs exact match, it must miss some similar words which is similar with the keywords. However, our scheme can make up for this disadvantage, and retrieve the most relevant files. Fig .2 shows that our method achieves remarkable result.

**Fig. 2.** Comparison of two schemes.

## 5 Conclusion

In this paper, a multi-keyword ranked search scheme over encrypted cloud data is proposed, which meanwhile supports latent semantic search. We use the vectors consisting of TF values as indexes to documents. These vectors constitute a matrix, from which we analyze the latent semantic association between terms and documents by LSA. Taking security and privacy into consideration, we employ a secure splitting k-NN technique to encrypt the index and the queried vector, so that we can obtain the accurate ranked results and protect the confidence of the data well. The experimental effect is remarkable

**Acknowledgments.** This work is supported by the NSFC (61232016, 61173141, 61173142, 61173136, 61103215, 61373132, 61373133), GYHY201206033, 201301030, 2013DFG12860, BC2013012 and PAPD fund.

## References

1. Armbrust, M., et al., A view of cloud computing. Communications of the ACM, 2010. **53**(4): p. 50-58.
2. Chuah, M. and W. Hu. Privacy-aware bedtree based solution for fuzzy multi-keyword search over encrypted data. in Distributed Computing Systems Workshops (ICDCSW), 2011 31st International Conference on. 2011. IEEE.
3. Deshpande, S., et al., Fuzzy keyword search over encrypted data in cloud computing. World Journal of Science and Technology, 2013. **2**(10).
4. Wang, C., et al. Secure ranked keyword search over encrypted cloud data. in Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on. 2010. IEEE.
5. Deerwester, S.C., et al., Indexing by latent semantic analysis. JASIS, 1990. **41**(6): p. 391-407.
6. Wong, W.K., et al. Secure kNN computation on encrypted databases. in Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. 2009. ACM.
7. Yang, C., et al. A Fast Privacy-Preserving Multi-keyword Search Scheme on Cloud Data. in Cloud and Service Computing (CSC), 2012 International Conference on. 2012. IEEE.
8. Powers, D.M. The problem with kappa. in Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. 2012. Association for Computational Linguistics.