Volume 2, Issue 10, October 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Secured Multi-keyword Ranked Search over Encrypted Cloud Data

Ankatha Samuyelu Raja
PG Student, CSE Department&CMRIT Hyderabad, India

Vasanthi A
CSE Department&CMRIT, Hyderabad, India

Abstract— Next generation computing stared with the advent of Cloud computing. In cloud computing data possessor are goaded to farm out their complex data management systems from local sites to the commercial public cloud for great flexibility and economic savings. To ensure the safety of stored data, it becomes must to encrypt the data before storing. In cloud the data search arises only with the plain data. But it is essential to invoke search with the encrypted data also. The specialty of cloud data storage should allow copious keywords in a solitary query and results the data documents in the relevance order. This paper focuses on multi keyword search based on ranking over an encrypted cloud data (MRSE). The search uses the feature of similarity and inner product similarity matching. The experimental results show that the overhead in computation and communication are considerably low.

Keywords— Cloud computing, Encrypted data, Multi keyword search, Ranked Search, Similarity Matching.

I. INTRODUCTION

Cloud computing is a term used to describe a set of IT services that are provided to a customer over a network on a leased basis and with the ability to scale up or down their service requirements. Clouds are large pools of easily usable and accessible virtualized resources. These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing optimum resource utilization. It's a pay-per-use model in which the Infrastructure Provider by means of customized Service Level Agreements (SLAs)[1] offers guarantees typically exploiting a pool of resources. Organizations and individuals can benefit from mass computing and storage centers, provided by large companies with stable and strong cloud architectures.

To protect data privacy and combat uninvited accesses in the cloud and further than, thin skinned data, e.g., emails, personal health records, photo albums, tax documents, financial transactions, etc., may have to be encrypted by data proprietor before outsourcing to the commercial public cloud [2]. This however, obsoletes the traditional data utilization service based on plaintext keyword search. The trivial solution of downloading all the data and decrypting locally is clearly impractical, due to the huge amount of bandwidth cost in cloud scale systems. Moreover, aside from eliminating the local storage management, storing data into the cloud serves no purpose unless they can be easily searched and utilized. Thus, exploring privacy-preserving and effective search service over encrypted cloud data is of paramount importance. Considering the potentially large number of on-demand data users and huge amount of outsourced data documents in the cloud, this problem is particularly challenging as it is extremely difficult to meet also the requirements of performance, system usability and scalability.

Need for data retrieval is the most frequently occurring task in cloud by the user to the server. The retrieval of the data should be fast enough. But the large amount of data space is used by the user, which in turn increases the time of search. Generally cloud server assigns ranks to document in order to make the search as faster. Such ranked search system enables data users to find the most relevant information quickly, rather than burdensomely sorting through every match in the content collection [3].Ranked Search can also elegantly eliminate unnecessary network traffic by sending back only the most relevant data, which is highly desirable in the "pay-as-you- use" cloud paradigm. For privacy protection, such ranking operation, however, should not leak any keyword related information. On the other hand, to improve the search result accuracy as well as to enhance the user searching experience, it is also necessary for such ranking system to support multiple keywords search, as single keyword search often yields far too coarse results. As a common practice indicated by today's web search engines (e.g., Google search), data users may tend to provide a set of keywords instead of only one as the indicator of their search interest to retrieve the most relevant data. And each keyword in the search request is able to help narrow down the search result further. "Coordinate matching" [4], i.e., as many matches as possible, is an efficient similarity measure among such multi-keyword semantics to refine the result relevance, and has been widely used in the plaintext information retrieval (IR) community. However, how to apply it in the encrypted cloud data search system remains a very challenging task because of inherent security and privacy obstacles, including various strict requirements like the data privacy, the index privacy, the keyword privacy, and many others.

This paper focuses on to the solution of multi-keyword ranked search over encrypted cloud data (MRSE) while preserving strict system-wise privacy in the cloud computing paradigm. A variety of multi- keyword semantics are available, an efficient similarity measure of "coordinate matching", i.e., as many matches as possible, to capture the relevance of data documents to the search query is used. Particularly "inner product similarity" [4], i.e., the number

of query keywords appearing in a document, to quantitatively evaluate such similarity measure of that document to the search query is used in MRSE algorithm.

II. PROBLEM DESCRIPTION

A. System Model

Cloud data storage service involves three actors 1. Data proprietor, Data User and Cloud Server as shown in Fig. 1. Data proprietor stores a set of document 'D' on to the cloud server in an encrypted form to avoid the security threats. To enable fast and cost effective data retrieval a search index 'I' is built over an encrypted data. To make a search, a set of keywords 'K' is given by an authorized user. The results are ranked using the ranking algorithm by the cloud server.

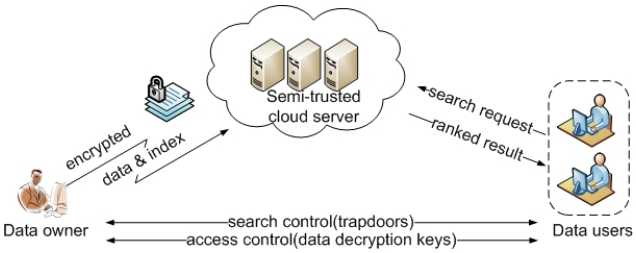

Fig. 1: Architecture of the search over encrypted cloud data

B. Design Objective

To activate the ranked search for effective utilization of outsourced cloud data under the aforementioned model, the system should be designed by considering the security considerations also. The system is expected to give the following security and performance guarantees as follows.

- Multi-keyword Ranked Search: To design search schemes which allow multi-keyword query and provide result similarity ranking for effective data retrieval, instead of returning undifferentiated results.
- Privacy-Preserving: To prevent the cloud server from learning additional information from the dataset and the index, and to meet the basic privacy requirements.
- Efficiency: Ranked search should ensure privacy and also low communication and computation overhead .

III. MRSE FRAMEWORK

For easy presentation, operations on the data documents are not shown in the framework since the data owner could easily employ the traditional symmetric key cryptography to encrypt and then outsource data. With focus on the index and query, the MRSE system consists of four algorithms as follows

1. ***Setup(ℓ)***

   Taking a security parameter ℓ as input, the data owner outputs a symmetric key as SK.

2. ***BuildIndex(F, SK)***

Based on the dataset F, the data owner builds a searchable index I which is encrypted by the symmetric key SK and then outsourced to the cloud server. After the index construction, the document collection can be independently encrypted and outsourced.

3. ***Trapdoor(fW)***

With t keywords of interest in fW as input, this algorithm generates a corresponding trapdoor TfW.

4. ***Query(TfW, k, I)***

When the cloud server receives a query request as (TfW, k), it performs the ranked search on the index I with the help of trapdoor TfW, and finally returns FfW, the ranked id list of top-k documents sorted by their similarity with fW.

.

A. Security and Privacy Requirements for MRSE Framework

The representative privacy guarantee in the related literature, such as searchable encryption, is that the server should learn nothing but search results. With this general privacy description, we explore and establish a set of strict privacy requirements specifically for the MRSE framework. As for the data privacy, the data owner can resort to the traditional symmetric key cryptography to encrypt the data before outsourcing, and successfully prevent the cloud server from prying into the outsourced data. With respect to the index privacy, [9][10] if the cloud server deduces any association between keywords and encrypted documents from index, it may learn the major subject of a document, even the content of a short document. Therefore the searchable index should be constructed to prevent the cloud server from performing such kind of association attack. While data and index privacy guarantees are demanded by default in the related literature, various search privacy requirements involved in the query procedure are more complex and difficult to tackle as follows.

Keyword Privacy:

As users usually prefer to keep their search from being exposed to others like the cloud server, the most important concern is to hide what they are searching, i.e., the keywords indicated by the corresponding trapdoor. Although the trapdoor can be generated in a cryptographic way to protect the query keywords, the cloud server could do some statistical analysis over the search result to make an estimate [15]. As a kind of statistical information, *document*

frequency (i.e., the number of documents containing the keyword) is sufficient to identify the keyword with high probability. When the cloud server knows some background information of the dataset, this keyword specific information may be utilized to reverse-engineer the keyword.

Trapdoor Unlinkability:

The trapdoor generation [5][6] function should be a randomized instead of being deterministic. In particular, the cloud server should not be able to deduce the relationship of any given trapdoors, e.g., to determine whether the two trapdoors are formed by the same search request. Otherwise, the deterministic trapdoor generation would give the cloud server advantage to accumulate frequencies of different search requests regarding different keyword(s), which may further violate the aforementioned keyword privacy requirement. So the fundamental protection for trapdoor unlinkability is to introduce sufficient no determinacy into the trapdoor generation procedure.

Access Pattern:

Within the ranked search, the access pattern is the sequence of search results where every search result is a set of documents with rank order. Specifically, the search result for the query keyword set fW is denoted as FfW, consisting of the id list of all documents ranked by their relevance to fW. Then the access pattern is denoted as (FfW1 ,FfW2 , . . .) which are the results of sequential searches. Although a few searchable encryption works, e.g., [11] has been proposed to utilize private information retrieval (PIR) technique [28], to hide the access pattern, our proposed schemes are not designed to protect the access pattern for the efficiency concerns. This is because any PIR based technique must "touch" the whole dataset outsourced on the server which is inefficient in the large scale cloud system.

In our more advanced design, instead of simply removing the extended dimension in the query vector as we plan to do at the first glance, we preserve this dimension extending operation but assign a new random number t to the extended dimension in each query vector. Such a newly added randomness is expected to increase the difficulty for the cloud server to learn the relationship among the received trapdoors. In addition, as mentioned in the keyword privacy requirement ,[7] randomness should also be carefully calibrated in the search result to obfuscate the document frequency and diminish the chances for re-identification of keywords. Introducing some randomness in the final similarity score is an effective way towards what we expect here. More specifically, unlike the randomness involved in the query vector, we insert a dummy keyword into each data vector and assign a random value to it. Each individual vector Di is extended to (n+2)-dimension instead of (n + 1), where a random variable εi representing the dummy keyword is stored in the extended dimension.

The whole scheme to achieve ranked search with multiple keywords over encrypted data is as follows.

1. Setup The data owner randomly generates a (n + 2)-bit vector as S and two (n+2)$\times$(n+2) invertible matrices $\{M1,M2\}$. The secret key SK is in the form of a 3-tuple as $\{S,M1,M2\}$.
2. BuildIndex(F, SK) The data owner generates a binary data vector Di for every document Fi, where each binary bit $Di[j]$ represents whether the corresponding keyword Wj appears in the document Fi. Subsequently, every plaintext subindex $\vec{D}\,i$ is generated by applying dimension extending and splitting procedures on Di. These procedures are similar with those in the secure kNN computation except that the (n + 1)-th entry in $\vec{D}\,i$ is set to a random number εi, and the (n + 2)-th entry in $\vec{D}\,i$ is set to 1 during the dimension extending. $\vec{D}\,I$ is therefore equal to (Di, εi, 1). Finally, the subindex Ii = $\{M^T_1\ \vec{D}_i{}',M^T_2\ \vec{D}_i{}''\}$ is built for every encrypted document Ci.
3. Trapdoor(fW) With t keywords of interest in fW as input, one binary vector Q is generated where each bit $Q[j]$ indicates whether $Wj \in$ fW is true or false. Q is first extended to n + 1-dimension which is set to 1, and then scaled by a random number $r \neq 0$, and finally extended to a (n + 2)-dimension vector as $\vec{Q}$ where the last dimension is set to another random number t. $\vec{Q}$ is therefore equal to (rQ, r, t). After applying the same splitting and encrypting processes as above, the trapdoor TfW is generated as $\{M^{-1}1\ \vec{Q}',M^{-1}2\ \vec{Q}''\}$.
4. Query(TfW, k, I) With the trapdoor TfW, the cloud server computes the similarity scores of each document Fi as in equation 1. WLOG, we assume $r > 0$. After sorting all scores, the cloud server returns the top-k ranked id list FfW. Note that in the original case, the final score is simply rDi

B. Efficiency Analysis

1) Index Construction: To build a searchable sub index I_i for each document F_i in the dataset F, the first step is to map the keyword set extracted from the document F_i to a data vector D_i, followed by encrypting every data vector. The time cost of mapping or encrypting depends directly on the dimensionality of data vector which is determined by the size of the dictionary, i.e., the number of indexed keywords. And the time cost of building the whole index is also related to the number of sub index which is equal to the number of documents in the dataset. Fig. 2(a) shows that, given the same dictionary where $|W|$ = 4000, the time cost of building the whole index is nearly linear with the size of dataset since the time cost of building each sub index is fixed. Fig. 2(b) shows that the number of keywords indexed in the dictionary determines the time cost of building a sub index. The major computation to generate a sub index in MRSE I includes the splitting process and two multiplications of a (n + 2) $\times$ (n + 2) matrix and a (n + 2) *2)*

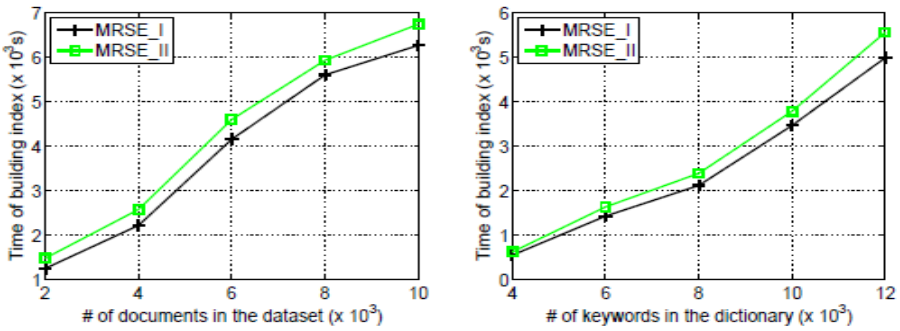

Fig. 2 a) Time cost of building Index for n=4000 b) Time cost of building Index for m=1000

Trapdoor Generation: Fig. 3(a) shows that the time to generate a trapdoor is greatly affected by the number of keywords in the dictionary. Like index construction, every trapdoor generation incurs two multiplications of a matrix and a split query vector, where the dimensionality of matrix or query vector is different in two proposed schemes and becomes larger with the increasing size of dictionary. Fig. 3(b) demonstrates the trapdoor generation cost in the MRSE II scheme is about 20 percentages larger than that in the MRSE I scheme. Like the sub index generation, the difference of costs to generate trapdoors is majorly caused by the different dimensionality of vector and matrices in the two MRSE schemes. More importantly, it shows that the number of query keywords has little influence on the overhead of trapdoor generation, which is a significant advantage over related works on multi-keyword searchable encryption.

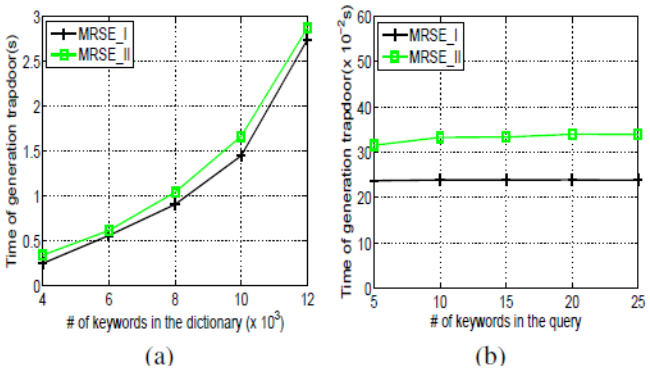

Fig. 3 (a) Time cost of generating trap door for t= 10 3 (b) for different number of query keywords n= 4000

3) Query: Query execution in the cloud server consists of computing and ranking similarity scores for all documents in the dataset. Fig. 6 shows the query time is dominated by the number of documents in the dataset while the number of keywords in the query has very slight impact on it like the cost of trapdoor generation above.

IV. CONCLUSION

In this paper, a new framework is proposed for the problem of multi-keyword ranked search over encrypted cloud data, and to establish a variety of privacy requirements. Among various multi-keyword semantics, the efficient similarity measure is "coordinate matching", i.e., as many matches are possible, to effectively capture the relevance of outsourced documents to the query keywords, and use "inner product similarity" to quantitatively evaluate such similarity measure. For meeting the challenge of supporting multi-keyword semantic without privacy breaches, MRSE framework is proposed using secure inner product computation. Thorough analysis investigating privacy and efficiency guarantees of proposed schemes is given, and experiments on the real-world dataset shows our proposed scheme introduces low overhead on both computation and communication.

REFERENCES

[1] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, 2009.
[2] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *RLCPS,January 2010, LNCS. Springer, Heidelberg*.
[3] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Engineering Bulletin*, vol. 24, no. 4, pp. 35–43, 2001.
[4] I.H.Witten, A.Moffat and T.C.Bell "Managing Gigabytes: Compressing and indexing documents and images", Morgan Kaughmann Publishing, San Fransisco, 1999.
[5] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. of S&P*, 2000.
[6] E.-J. Goh, "Secure indexes," Cryptology ePrint Archive, 2003, http:// eprint.iacr.org/2003/216.

[7] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proc. of ACNS*, 2005.

[8] R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in *Proc. of ACM CCS*, 2006.

[9] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. of EUROCRYPT*, 2004.

[10] M. Bellare, A. Boldyreva, and A. ONeill, "Deterministic and efficiently searchable encryption," in *Proc. of CRYPTO*, 2007.

[11] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency properties, relation to anonymous ib e, andextensions," *J. Cryptol.*, vol. 21, no. 3, pp. 350–391, 2008.

[12] [12] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proc. of IEEE INFOCOM'10 Mini-Conference*, San Diego, CA, USA, March 2010.

[13] [13] D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. E. S. III, "Public key encryption that allows pir queries," in *Proc. of CRYPTO*, 2007.

[14] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," in *Proc. of ACNS*, 2004, pp. 31–45.

[15] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," in *Proc. of ICICS*, 2005.

