ACHIEVING SECURE AND DYNAMIC RANGE QUERIES OVER ENCRYPTED CLOUDDATA

P.Vijay Bhaskar Reddy, Associate Professor, Dept.of Master of Computer Applications, Narayana Engineering College(Autonomous), Gudur.SPSR Nellore, AP, India
R.Akhila, PG Scholar, Dept.of Master of Computer Applications, Narayana Engineering College(Autonomous), Gudur.SPSR Nellore, AP, India

Abstract - Due to the increasing popularity of cloud computing, more and more data owners are motivated to outsource their data to cloud servers for great convenience and reduced cost in data management. However, sensitive data should be encrypted before outsourcing for privacy requirements, which obsoletes data utilization like keyword-based document retrieval. In this paper, we present a secure multi-keyword ranked search scheme over encrypted cloud data, which simultaneously supports dynamic update operations like deletion and insertion of documents. Specifically, the vector space model and the widely-used TF IDF model are combined in the index construction and query generation. We construct a special tree-based index structure and propose a "Greedy Depth-first Search" algorithm to provide efficient multi-keyword ranked search. The secure kNN algorithm is utilized to encrypt the index and query vectors, and meanwhile ensure accurate relevance score calculation between encrypted index and query vectors. In order to resist statistical attacks, phantom terms are added to the index vector for blinding search results. Due to the use of our special tree-based index structure, the proposed scheme can achieve sub-linear search time and deal with the deletion and insertion of documents flexibly. Extensive experiments are conducted to demonstrate the efficiency of the proposed scheme.

Index Terms - Searchable encryption, multi-keyword ranked search, dynamic update, cloud computing

1.INTRODUCTION

Cloud computing has been considered as a new model of enterprise IT infrastructure, which can organize huge resource of computing, storage and applications, and enable users to enjoy ubiquitous, convenient and on-demand network access to a shared pool of configurable computing resources with great efficiency and minimal eco-nomic overhead [1]. Attracted by these appealing features, both individuals and enterprises are motivated to outsource their data to the cloud, instead of purchasing software and hardware to manage the data themselves. Despite of the various advantages of cloud services, outsourcing sensitive information (such as e-mails, per-sonal health records, company finance data, government documents, etc.) to remote servers brings privacy con-cerns. The cloud service providers (CSPs) that keep the data for users may access users' sensitive information without authorization. A general approach to protect the data confidentiality is to encrypt the data before outsourc-ing [2]. However, this will cause a huge cost in terms of data usability. For example, the existing techniques on keywordbased information retrieval, which are widely used on the plaintext data, cannot be directly applied on the encrypted data. Downloading all the data from the cloud and decrypt locally is obviously impractical. In order to address the above problem, researchers have designed some general-purpose solutions with fully-homo-morphic encryption [3] or oblivious RAMs [4]. However, these methods are not practical due to their high computa-tional overhead for both the cloud sever and user. On the contrary, more searchable encryption (SE) schemes have made specific con-tributions in terms of efficiency, functionality and security. Searchable encryption schemes enable the client to store the encrypted data to the cloud and execute keyword search over ciphertext domain.

2.RELATED WORK

Searchable encryption schemes enable the clients to store the encrypted data to the cloud and execute keyword search over ciphertext domain. Due to different cryptography primitives,

searchable encryption schemes can be con-structed using public key based cryptography [5], [6] or symmetricbased cryptography [7], [8], [9], [10]. Song et al. [7] proposed the first symmetric searchable encryption (SSE) scheme, and the search time of their scheme is linear to the size of the data collection. Goh [8] proposed formal security definitions for SSE and designed a scheme based on Bloom filter. The search time of Goh's scheme is Oð nÞ, where n is the cardinality of the document collection. Curtmola et al. [10] proposed two schemes (SSE-1 and SSE-2) which achieve the optimal search time. Their SSE-1 scheme is secure against chosen-keyword attacks (CKA1) and SSE-2 is secure against adaptive chosenkeyword attacks (CKA2). These early works are single keyword boolean search schemes, which are very simple in terms of functionality. Afterward, abundant works have been proposed under

TD—The encrypted form of Q, which is named as trapdoor for the search request. Du—The index vector stored in tree node u whose dimension equals to the cardinality of the dictionary W. Note that the node u can be either a leaf node or an internal node of the tree. The encrypted form of Du.

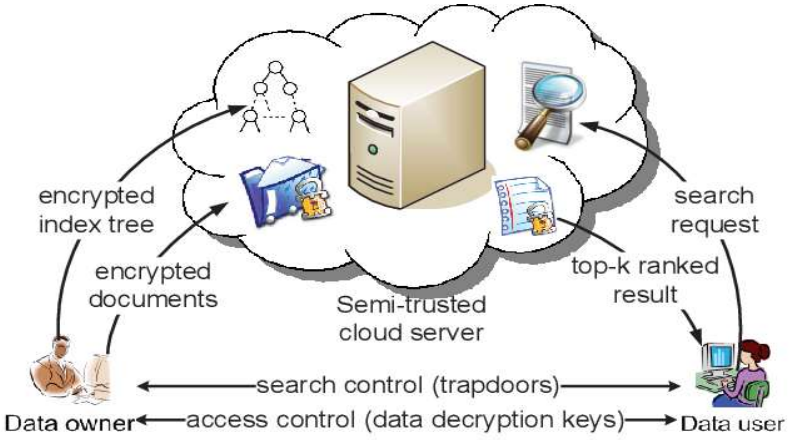


Fig. 1. The architecture of ranked search over encrypted cloud data

2.1Design Goals

To enable secure, efficient, accurate and dynamic multi-key-word ranked search over outsourced encrypted cloud data under the above models, our system has the following design goals.

2.2 Dynamic.

The proposed scheme is designed to provide not only multi-keyword query and accurate result **on** document collections**.** ranking, but also

2.3 Search efficiency.

The scheme aims to achieve sublinear search efficiency by exploring a special tree-based index and an efficient search algorithm.

2.4 Privacy-preserving. The scheme is designed to prevent the cloud server from learning additional information about the document collection, the index tree, and the query. The spe-cific privacy requirements are summarized as follows,

1) Index confidentiality and query confidentiality. The underlying plaintext information, including key-words in the index and query, TF values of key-words stored in the index, and IDF values of query keywords, should be protected from cloud server;

2) Trapdoor unlinkability. The cloud server should not be able to determine whether two encrypted queries (trapdoors) are generated from the same search request;

3) Keyword privacy. The cloud server could not identify the specific keyword in query, index or document collection by analyzing the statistical information like term frequency. Note that our proposed scheme is not designed to protect access pattern, i.e., the sequence of returned documents.

3.THE PROPOSED SCHEMES

In this section, we first describe the unencrypted dynamic multi-keyword ranked search (UDMRS) scheme which is constructed on the basis of vector space model and KBB tree. Based on

the UDMRS scheme, two secure search schemes (BDMRS and EDMRS schemes) are constructed against two threat models, respectively.

3.1 Index Construction of UDMRS Scheme In Section 3, we have briefly introduced the KBB index tree structure, which assists us in introducing the index con-struction. In the process of index construction, we first gen-erate a tree node for each document in the collection. These nodes are the leaf nodes of the index tree. Then, the internal tree nodes are generated based on these leaf nodes. The for-ma

construction process of the index is presented in Algorithm 1. An example of our index tree is shown in Fig. 3. Note that the index tree T built here is a plaintext.

3.2 Search Process of UDMRS Scheme The search process of the UDMRS scheme is a recursive pro-cedure upon the tree, named as "Greedy Depth-first Search" algorithm. We construct a result list denoted as RList, whose element is defined as hRScore; FIDi. Here, the RScore is the relevance score of the document fFID to the query, which is calculated according to Formula (1). The RList stores the k accessed documents with the largest rele-vance scores to the query. The elements of the list are ranked in descending order according to the RScore, and will be updated timely during the search process. Since the possible largest relevance score of documents rooted by the node u can be predicted, only a part of the nodes in the tree are accessed during the search process.

BDMRS Scheme Based on the UDMRS scheme, we construct the basic dynamic multi-keyword ranked search scheme by using the secure kNN algorithm [38]. The BDMRS scheme is designed to achieve the goal of privacypreserving in the known ciphertext model, and the four algorithms included are described as follows: SK SetupðÞ Initially, the data owner generates the secret key set SK, including 1) a randomly generated m-bit vector S where m is equal to the cardinality of dictionary, and 2) two ðm mÞ invertible matrices M1 and M2. If there is no idle processor, the current processor is used to deal with the child with larger rele-vance score, and the other child is put into a waiting queue. Once there is an idle processor, it takes the oldest node in the queue to continue the search. Note that all the process-ors share the same'precision' is defined as that in [26]: Pk ¼ k0=k, where k0 is the number of real top-k documents in the retrieved k documents. If a smaller standard deviation s is set for the random variable P "v, the EDMRS scheme is supposed to obtain higher pre-cision, and vice versa. The results are shown in Fig. 4a. In the EDMRS scheme, phantom terms are added to the index vector to obscure the relevance score calculation, so that the cloud server cannot identify keywords by analyzing the TF distributions of special keywords. Here, we quantify the obscureness of the relevance score by "rank privacy", which is defined as: where r0 i is the rank number of document in the retrieved top-k documents, and r0i is its real rank number in the whole ranked results. The larger rank privacy denotes the higher security of the scheme, which is illustrated in Fig. 4b. In the proposed scheme, data users can accomplish dif-ferent requirements on search precision and privacy by adjusting the standard deviation s, which can be treated as a balance parameter.

IV. PERFORMANCE ANALYSIS

We implement the proposed scheme using C++ language in Windows 7 operation system and test its efficiency on a real-world document collection: the Request for Comments (RFC) [39]. The test includes 1) the search precision on dif-ferent privacy level, and 2) the efficiency of index construc-tion, trapdoor generation, search, and update. Most of the experimental results are obtained with an Intel Core(TM) Duo Processor (2.93 GHz), except that the efficiency of search is tested on a server with two Intel(R) Xeon(R) CPU E5-2620 Processors (2.0 GHz), which has 12 processor cores and supports 24 parallel threads

4.1 Precision and Privacy The search precision of scheme is affected by the dummy keywords in EDMRS scheme. Here, the'precision' is defined as that in [26]: Pk ¼ k0=k, where k0 is the number of real top-k documents in the retrieved k documents. If a smaller standard deviation s is set for the random variable P "v, the EDMRS scheme is supposed to obtain higher pre-cision, and vice versa. The results are shown in diagram.

4.2 Efficiency

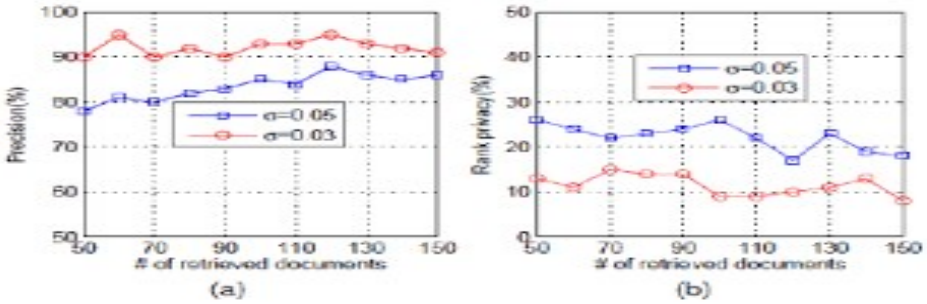
4.2.1 Index Tree Construction

The process of index tree construction for document collection F includes two main steps: 1) building an unencrypted KBB tree based on the document collection F , and 2) encrypting the index tree with splitting operation and two multiplications of matrix. The index structure is constructed following a post order traversal of the tree based on the document collection F , and nodes are generated during the traversal. For each node, generation of an index vector takes time, vector splitting process takes time, and two multiplications of a matrix takes time. As a whole, the time complexity for index tree construction is Oðnm2Þ. Apparently, the time cost for building index tree mainly depends on the cardinal-ity of document collection F and the number of keywords in dictionary W. Fig. 5 shows that the time cost of index tree construction is almost linear with the size of document col-lection, and is proportional to the number of keywords dictionary. Due to the dimension extension, the index tree construction of EDMRS scheme is slightly more time-consuming than that of BDMRS scheme. Although the index tree construction consumes relatively much time at the data owner side, it is noteworthy that this is a one-time operation. On the other hand, since the underlying balanced binary tree has space complexity OðnÞ and every node stores two m-dimensional vectors, the space complexity of the index tree is OðnmÞ. As listed in Table 3, when the document collection is fixed (n ¼ 1;000), the storage consumption of the index tree is determined by the size of the dictionary.

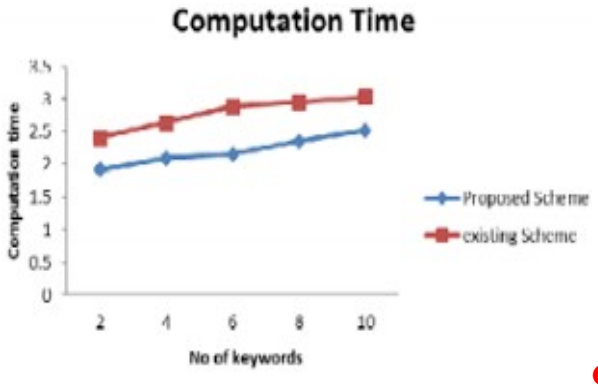
4.2.2 Trapdoor Generation The generation of a trapdoor incurs a vector splitting opera-tion and two multiplications of a ðm mÞ matrix, thus the time complexity is Oðm2 Þ, as shown in Fig. 6a. Typical search requests usually consist of just a few keywords. Fig. 6b shows that the number of query keywords has little influence on the overhead of trapdoor generation when the dictionary size is fixed. Due to the dimension extension, the time cost of EDMRS scheme is a little higher than that of BDMRS scheme.

4.2.3 Search Efficiency During the search process, if the relevance score at node u is larger than the minimum relevance score in result list RList, the cloud server examines the children of the node; else it returns. Thus, lots of nodes are not accessed during a real search. We denote the number of leaf nodes that contain one or more keywords in the query as u. Generally, u is larger than the number of required documents k, but far less than the cardinality of the document collection n. As a balanced binary tree, the height of the index is maintained to be log n.

Storage consumption of table tree



5.REFERENCES



1. www.w3schools.com
2. www.cooltext.com
3. www.msdn.com
4. Facebook, http://www.yahooanswer.com.
5. Twitter, http://twitter.com.
6. Foursquare http://www.foursquare.com.
7. Google latitude, http://www.google.com/intl/enus/latitude/intro.html.
8. Buddycloud, http://buddycloud.com.
9. Mobile instant messaging, http://en.wikipedia.org/wiki/Mobile instant messaging.
10. R. B. Jennings, E. M. Nahum, D. P. Olshefski, D. Saha, Z.-Y.Shae, and C. Waters, ”A study of internet instant messaging and chat protocols,” IEEE Network, 006.
11. Gobalindex, http://www.skype.com/intl/en-us/support/user-guides/p2pexplained/