ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 2, February 2015

Reusability of Search Index over Encrypted Cloud Data on Dynamic update

Kavitha R[1], R J Poovaraghan[2]

Student, M.Tech, SRM University, Chennai, India[1]

Assistant Professor (OG), Department of Computer Science, SRM University, Chennai, India[2]

Abstract: Cloud computing is generating lot of interest to provide solution for data outsourcing and high quality data services. More and more institution, organizations and corporations are exploring the possibility of having their applications, data and their IT assets in cloud. As the data and there by the cloud's size increases searching of the relevant data is expected to be a challenge. To overcome this challenge, search index is created to aid in faster search. However, search Index creation and computation has been complex and time consuming, leading to cloud-down time there by hindering the swiftness in reacting to data request for mission critical requirements.
Focus of this paper is to explain how in the proposed system, reusability of search index is helping to reduce the complexity of search index computation. Search index is proposed to be created using parameters like similarity relevance, user ranking and scheme robustness. User ranking helps to guarantee why a phrase or a sentence or a key word is used frequently in the uploaded data. The proposed system ensures that the reusability of search index concept, highly reduces cloud down time while maintaining the security using searchable symmetric encryption (SSE).The user requested file is retrieved from the cloud, using Two-round searchable encryption (TRSE) scheme that supports top-k multi-keyword retrieval.

Keywords: Cloud, data privacy, search index, ranking, homomorphic encryption

I. INTRODUCTION

Cloud computing [1] is a recently evolved computing terminology or metaphor based on utility and consumption of computing resources. Cloud computing involves deploying groups of remote servers and software networks that allow centralized data storage and online access to computer services or resources. Though cloud computing promises to be a powerful technique to help users and user organizations reduce their infrastructure costs / improve efficiency, concerns regarding the security of the data and the data loss [3], has withheld organizations from embracing the technology whole heartedly.
To address this concern, users tend to encrypt their data on the cloud using advanced encryption algorithms. In cloud computing, data owners may share their data in the cloud with authorized users who in turn might want to retrieve only the data files they are interested in. Availability of required data at the right time and in the right format will be a key factor for gaining the acceptance of the end user.Currently, keyword based retrieval is one of the most popular ways to retrieve the file in the cloud. Earlier, there were methods or SSE schemes [4][5] which support only Boolean keyword search. i.e., whether a keyword exists in a file or not. Later on the key word search was enhanced to include multiple-keywords.
In multi keyword based search index using top-k[6][7][8], user Ranking plays an important factor. User ranking guarantee why something is mentioned a lot. Search index is created for the files based on the user ranking. User ranking [secure][9] is an input to the cloud server and the retrieval of relevant files/data is performed by the cloud server depending on the ranking and the relevance score of the respective files. Homomorphic encryption scheme is used to create a secure search index which is provided along with the encrypted files to the server. Cloud server uses TRSE encryption scheme which ensures security over the data. In a practical cloud computing system, data update like adding or deleting files leads to a new challenge to searchable encryption scheme. For ex., with proliferation of data, data update may become frequent for newly added files and the frequency depends on the business need. In these scenarios, it is necessary to update the search index for the new files such that the retrieval of those may happen smoothly without any cloud downtime or any performance degradation. The search Index computation for addition of each and every file takes considerable amount of time and also may lead to some overhead costs and communication when the cloud is outsourced.

Tablet
Notebook
Documents
Training
Database
Presentations
Calendar
Collaboration
Finance
Storage
Spreadsheets
Contacts
Email
Smartphone
Desktop

Figure 1 - Cloud Server

This paper introduces the concept of reusability of the search index, when dynamic updates like add, delete or update of a file / new file addition to the cloud server happens.

The Contributions can be summarized asfollows:

➔ The search index of the frequently searched words is maintained in the primary index and is re-used when any new files are getting added to the same category or group.

➔ The frequently used words are scanned in the incoming file to the cloud server and if it approximately matches the existing search index, the same index is shared between the files instead of creating a new one

➔ The search index for the low frequency words is maintained in a secondary index and can be moved to primary index if the frequency of usage increases

➔ The search index of the deleted files are retained for a limited period of time during which a new file with similar keywords are uploaded, then the same search index is mapped for the new file

The search index reusability ensures the cloud downtime is reduced effectively. The benefits could be large when applied to thousands of files entering the cloud server and in multi-cloud architecture where multiple clouds participate.

II. EXISTING SYSTEM

Existing system consists of three modules. Data Owner, Cloud server and Cloud User.

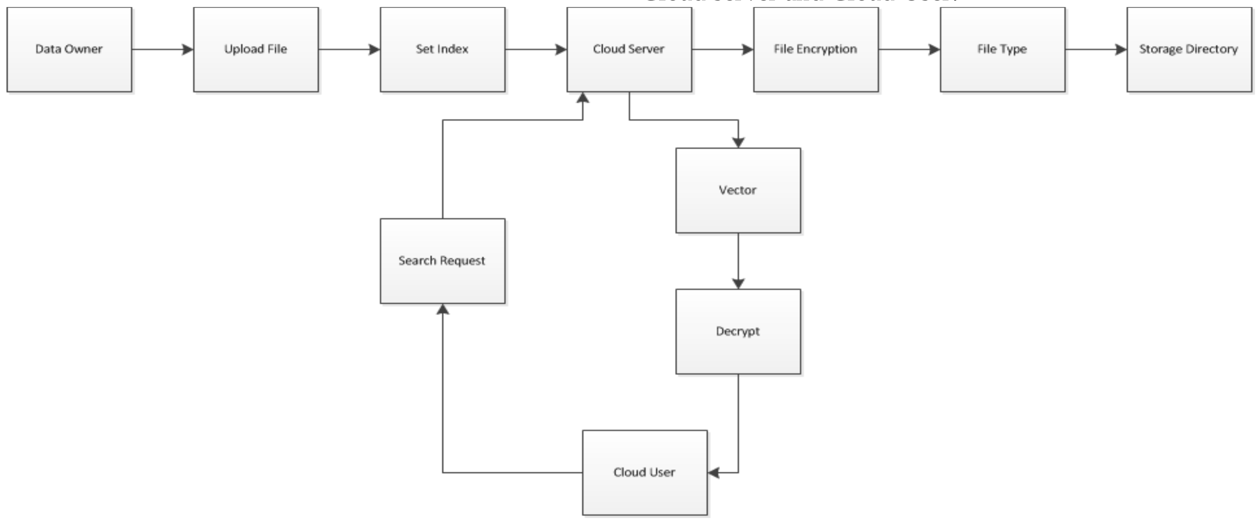

Figure 2 - Existing System Architecture

➔ Data Owner has X number of files to be uploaded to the cloud server. He encrypts the input files and the search index he creates for the file before uploading to the server.

▪ The search index is calculated based on the parameter value (tf-idf) and its relevance score

▪ Tf ($tf_{t,f}$) – Term Frequency : denotes the number of occurrences of the term t, in file f

▪ Dft –Document frequency : refers to the number of files that contains the term t

▪ Idft – Inverse document frequency is defined as: $idf_t=\log(N/df_t)$, N denotes the total number of files

The tf-idf weighting scheme assigns to a term t a weight in file f given by

$\text{tf-idf}_{t,f}=tf_{t,f} \times idf_t$

The weights of the term that occur very frequently in the collection are diminished and the weights of the terms that occur rarely are increased [2].Tf-idf depicts the weight of a single keyword on a file, to specify multiple keywords vector space model is employed. Vector space model [12] is an algebraic model for representing file as a vector. Each dimension of a vector corresponds to a separate term. The vector space model supports multi-term and non-binary distribution[2].

Homomorphic encryption is a scheme that allows specific types of computations to be carried out on the corresponding cipher text. The result is the cipher text of the result of the same operations performed on the plaintext. The encryption is modified to reduce the size of the cipher textwithout altering its homomorphism property.This method is used to encrypt search index and it involves four stages: KeyGen, Encrypt, Evaluate, and Decrypt. However, the actual files are encrypted using one of the symmetric encryption schemes, since the homomorphic encryption scheme is time consuming.

➔ Cloud server stores the files and its respective index in the data storage. When the data user requests with a multi-keyword query, it searches for the relevant files based on the keyword and provides the user with the terms and its ranking. The data user decrypts and requests for the highly scored files which in turn is provided by the server. The cloud server uses two round TRSE scheme for this purpose.

➔ Data user requests for files in the cloud server using a multi-keyword query,which is retrieved based on ranking of the keyword in the search index.

A. Disadvantages

➔ Search Index need to be created dynamically every time a file is updated and it involves a complex process since homomorphic encryption is used.This encryption scheme for search index is a time consuming one as KeyGen process involves creating secret key and public key every time for encryption and decryption.

➔ Calculating Search index for every incoming file involves high user computation and is not efficient in a multi user / multi file scenario

➔ Resources currently employed by companies to create search index could be used in more productive ways

III. PROPOSED SYSTEM

Proposed system concentrates on solving the disadvantages in the existing system by reusing and sharing the search index. This system can also be employed in large industries where data owners and cloud servers are trusted.

The cloud server hosts third-party data and retrieve the stored data. Since data may contain sensitive information, the cloud servers cannot be fully entrusted in protecting data. The data owner uploads 'N' number of files and expects cloud server to provide keyword retrieval service to authorized users.

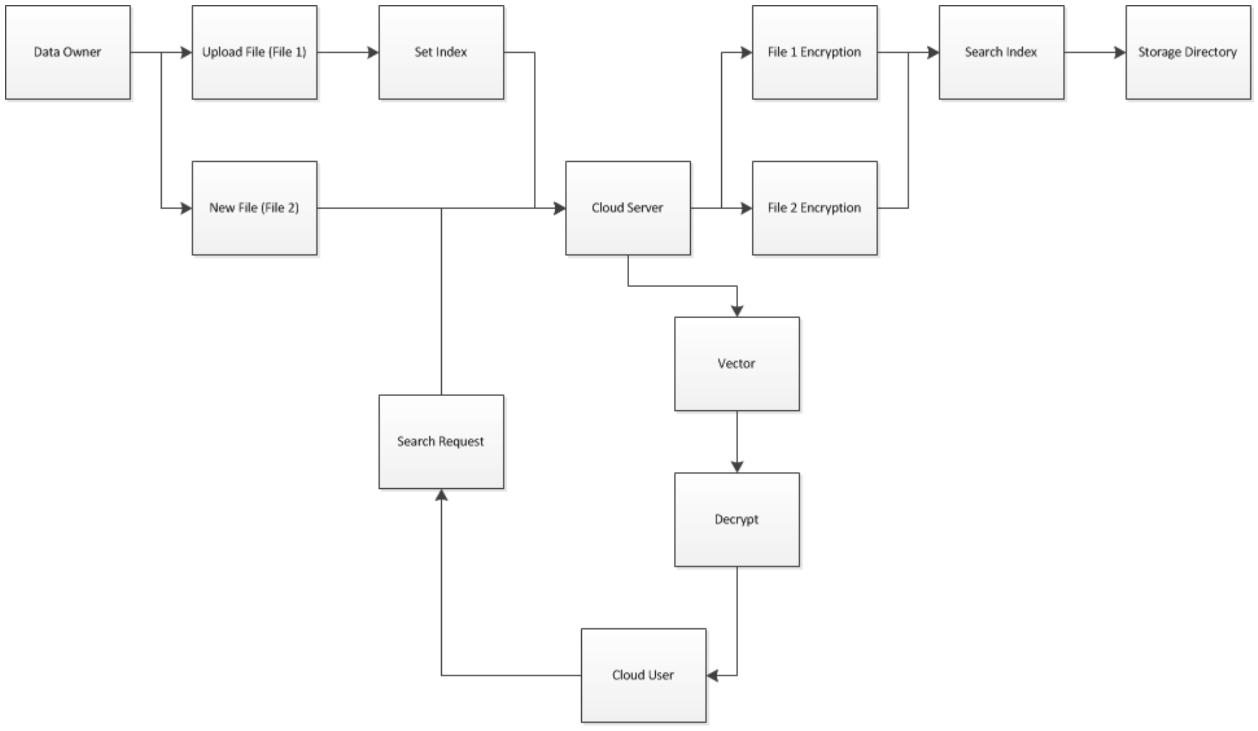

Figure 3 - Architecture of the proposed system

The proposed system comprises of the following modules.

A. *Creation of search index*

➔ The data owner creates the search index to calculate relevance factor which depends on user ranking

➔ Tf (term Frequency), Dft (Document frequency) and Idft (Inverse document frequency) are included in the calculation of tf-idf=$tf_{t,f}$ X idftvalues

➔ Vector space model [12] is used for representing multiple keywords as in the current system

➔ The search index is encrypted using homomorphic encryption which allows complex mathematical operations to be performed on encrypted data and it includes four stages:

▪ KeyGen : The secret key and the public key are generated for encryption and decryption

▪ Encrypt : Cipher text is generated with the key generated from the KeyGen

▪ Evaluate : The binary addition and multiplication is applied to compute tf-idf

▪ Decrypt: Method that satisfies homomorphism property

B. *Re-using search index for the dynamic update of files*

The drawback of the current system emerges when adding/removing of files to and from the cloud server. Search Index needs to be calculated for each entry of the file and the computation complexity of the search index is a time consuming activity. To avoid this, the proposed system plans reusing of the existing search index.

➔ Addition of new files to the server(same domain) :

▪ Check for the type/domain of the input files (e.g., security/storage)

▪ The cloud server checks for the frequently occurring words in the document

▪ Compare the keywords with the already existing Search index

▪ If the words match, the same ranking/value could be re-used for the new file. This is achieved since the cloud server retrieves the file based on relevance criteria for the queried keywords

➔ Addition of new files to the server(different domain) :

▪ If the input file is completely different from the existing domains in the server, the cloud server could request the data owner to compute the search index

➔ The highly searched keywords by the end user or data user is maintained or tracked in a file

C. Plan when a file is deleted

➔ If a particular file that comprises of frequently used keywords is removed from the server, then the search index for the removed/deleted file can be maintained in the server for future use

➔ The retention period to preserve the search index for the high frequent words depends on the business need

➔ Primary and Secondary indices plays an important role here. While primary indices store the search index of the frequently used words in the immediate past (period depends on business need), the words that are not used frequently in the same period can be moved to secondary index

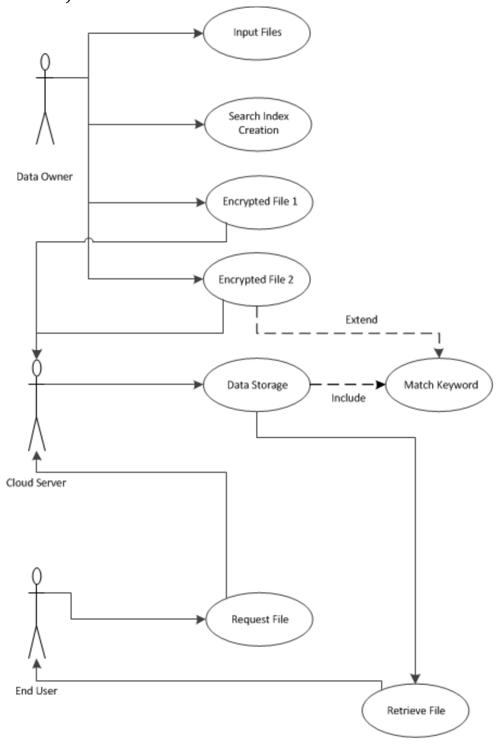

Figure 4 - Use Case Diagram

D. Advantages

➔ Sharing the search index for multiple files could effectively manage the space for search index in the cloud

➔ Cloud-down time could be reduced by re-using or sharing the search index which avoids the need for its frequent computation

➔ Resources used for computation can be diverted to other requirements.

IV. RELATED WORK

A Swaminathan et al. [7] explored secure rank-ordered retrieval with improved searchable encryption in the scenario of data center. They built a framework for privacy-preserving top k retrieval, including secure indexing and ranking.N.cao et al.[11] made the first attempt to define and solve the problem of top-k multi-keyword retrieval over encrypted cloud data. H.Hu et al. [10] employed homomorphism to preserve the data privacy. They devised a protocol for processing the nearest neighbor node using index query to preserve the data at both client and owner perspective but the query with multi-keyword is not supported. Jiadi Yu et al.[2] made an attempt to produce the cipher text with small size using modified homorphism. They built the search index using homorphism which provided security but had to update the search index on dynamic update of the files. In this paper, the same method is used for creating search index but here the focus is on reusability and sharing of the search index which provides business benefits.

V. CONCLUSION

In this paper, the problem of search index computation for every newly added file is resolved by reusing and sharing of the index which reduces resource requirement for search index computation. The sharing of search index also helps to manage the space effectively in the cloud server. Cloud down time will be made to zero when this technique is used effectively.The real business benefits are achieved in industries where large number of files are handled.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and M. Zaharia. "A view of cloud computing," Communication of the ACM 53(4): 50 58, 2010.

[2] Jiadi Yu, Peng Lu, Yanmin Zhu, GuangtaoXue and MingluLi , "Towards Secure Multi-Keyword Top-k Retrieval over Encrypted Cloud Data", 2013

[3] Amazon.com, "Amazon s3 availability event: July 20, 2008," http://status.aws.amazon.com/s3-20080720/html, 2008

[4] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proc. of IEEE Symposium on Security and Privacy, 2000

[5] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data,"in Proc. of IEEE Symposium on Security and Privacy, 2000

[6] S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, "Zerber+r: Top-k retrieval from a confidential index," in Proc. of EDBT, 2009

[7] A. Swaminathan, Y. Mao, G.-M. Su, H. Gou, A. L. Varna, S. He, M. Wu, and D. W. Oard, Confidentiality-preserving rank-ordered search," in Proc. of the Workshop on Storage Security and Survivability, 2007

[8] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multikeyword ranked search over encrypted cloud data," in Proc. of IEEE INFOCOM, 2011

[9] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in Proc. of ICDCS, 2010

[10] H. Hu, J. Xu, C. Ren and B. Choi, "Processing private queries over untrusted data cloud through privacy homomorphism," in Proc. of ICDE, 2011

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

[12] D. Dubin, "The Most Influential Paper Gerard Salton Never Wrote," LIBRARY TRENDS, 2004