TELKOMNIKA, Vol. 11, No. 10, October 2013, pp. 5884 ~ 5889
ISSN: 2302-4046

Fuzzy Keyword Search with Safe Index over Encrypted Cloud Computing

Lixi Liu, Chi Zhang, Shaowen Yao, Shipu Wang, Wei Zhou*
National Software School of YunNan University, Yunnan, China
*Corresponding author, e-mail: wz.weizhou@gmail.com

Abstract

As cloud computing becomes more and more utilized, sensitive data are being stored in cloud central servers. To ensure privacy, these data are usually encrypted before uploading, which makes searching complicated. In this paper we propose architecture of fuzzy keyword search with safe index. Bloom filter is used to build safe index which increases the searching effectiveness on a large dataset. The size of index is fixed so that the space efficiency is promoted. The experiment result shows that our searching scheme is effective and can be adapted to large encrypted file system.

Keywords*: cloud computing, fuzzy keyword, safe index, Bloom filter*

1. Introduction

With recent significant development in the cloud computing, the number of research for how to change the architecture or the workflow for the tranditional application server systems on the basis of cloud computing increases quickly such as [1] which designs a new distributed application server system based on cloud computing. And at the same time people have further worry about cloud security. Some researches focus on the trust-control architecture of cloud computing or the safe scheme targetes some situations have be proposed during the recent years such as the scheme for the protection of the Personal Health Records in the cloud which is proposed by [2]. And in this paper we focus on the safe search over encrypted cloud.

The main idea of traditional searchable encryption schemes is to build an encrypted searchable index so that it does not leak any information to the server. Song et al. [3] firstly studied the traditional single keyword searchable encryption in the symmetric key setting. A special two-layered encryption construction was used to encrypt each word. Goh [4] improvements system efficiency by using Bloom filter to construct the index for each data file. Chang et al. [5] and Curtmola et al. [6] propose a symmetric key encryption approach that offers better efficiency. In the research of reducing data storage and communication cost Jin Wook Byun *et al.* [7] proposed a more efficient conjunctive keyword search.

With the increasing popularity of using cloud computing platform, there has been, in recent years, a growing interest in techniques related to searching on encrypted data. Jin Li[8] propose the scheme to do fuzzy keyword search over encrypted data in cloud computing. They construct Wildcard-based fuzzy set for the keywords before building index. It can deal with minor typos and format inconsistencies when users type in query keyword. Ning Caoy[9] define and solve the problem of multi-keyword ranked search over encrypted cloud data, and establish a variety of privacy requirements. To reduce the user's computational overhead, Qin Liu et al. [10] utilized the cloud provider to participate in the partial decipherment of the files.

In this paper we propose architecture of fuzzy keyword search. We use the fuzzy method of [8] to construct fuzzy keyword sets. This technique eliminates the demand for enumerating all the fuzzy keywords, meanwhile the size of the fuzzy keyword sets is controlled. Based on the constructed fuzzy keyword sets, we propose an efficient fuzzy keyword search scheme. Bloom filter is used to build search index which increases the searching effectiveness on a large dataset. The size of index is fixed so that the space efficiency is promoted. Furthermore the time complexity of searching in bloom filter is O(n), so the time efficiency is optimized. In the end through complete experiments we show that our scheme is secure and efficient.

Received April 22, 2013; Revised July 4, 2013; Accepted July 18, 2013

This paper is organized as follows: Section 2 briefly describes our system architecture; Section 3 presents bloom filter and the way we use it to build safe index; Section 4 states detailed procedures and system security; Section 5 shows the simulation results and analysis, and Section 6 concludes this paper.

2. Architecture

An architecture of system is shown in the Figure 1. Our system is divided into three parts: Data owner, Cloud server and Data searcher. When searcher send a request to the server, sever search its local index and send the result to the user. After receiving the query result, the user can download, modify or delete the file from the cloud server using the file identifier.

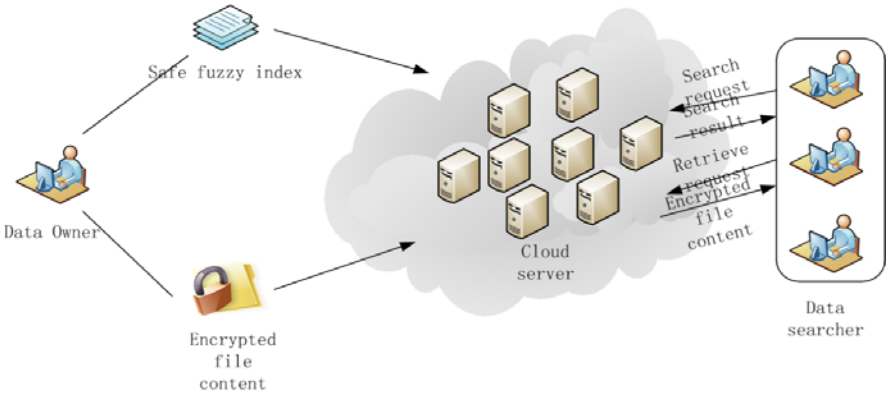

Figure 1. System Architecture

3. Safe Index by Bloom Filter

To guarantee the keywords privacy, all the keywords should be encrypted by user's private key, so that cloud servers have no knowledge of the content of queries. We use Bloom Filter to do the carriers of fuzzy keywords indexes. The goal of this is to ensure the security of the index and avoid information leaking.

3.1. Bloom Filter Basic

Bloom Filter is a random data structure which has high space efficiency. It represents a set of S={x1, x2, x3, ..., xn} of n elements and is represented by an array of m bits. All the bits are initialed to 0. Bloom Filter uses i independent hash functions h1, ..., hi. And the functions are used to set each position of the array 0 or 1. If xi belong to S, the array bits at the position h1(xi), h2(xi), ...,hi(xi) are set to 1. If one position is set to 1 multiple times, only the first time counts. To tell whether x belongs to S, we just need to calculate the values of the position h1(x), h2(x), ..., hi(x). If each value is 1, then x has high possibility to be the element of S. If any checked bits are 0, then x is definitely not a member of S. The procedure is showed by Figure 2.

3.2. Building Algorithm

In our scheme, a safe index is built to guarantee the privacy during searching. Here we give a process about how to build safe index.

Firstly the keyword is used as input. Then we use the master key to encrypt the coded keyword called trapdoor. Secondly trapdoor combining with the file identifier are used as input. We use bloom filter to build the safe index with the combination of trapdoors and file IDs. Also the file IDs are encrypted.The method about building safe index by bloom filter is proposed firstly by [4] and in our scheme we do some improvements in the implementation. Pseudo-random functions are used in the process both in Keyword Trapdoor Generation and Index Trapdoor Generation. Twice different pseudo-random procession can totally eliminate keyword information. The safe index is described below.

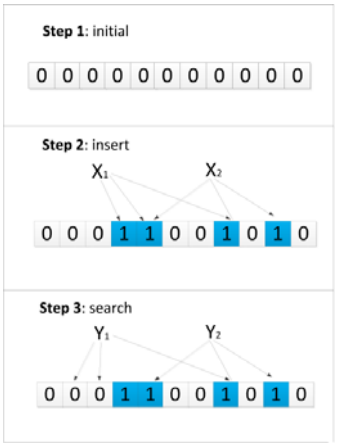

Figure 2. Bloom Filter basic

Table 1. Parameters in the steps

Function Variable	Meaning
mkey	The master key which is owned by the server
FN	Name of the file
word	Word in the fuzzy keyword set
Key	The key which is used to encrypted the word
Fid	Generated by using the file name
f1(x)	One of the hash functions
Keyword Trapdoor	The encrypted keyword
f2(x)	The other hash function
Index Trapdoor	Generated by using keyword trapdoor and Fid
Bloom filter Index	The index build in bloom filter

1) Key Generate (mkey): Use the master key to generate a private key.This step is to ensure the safety of user's key.
2) Fid Generate (FN): Use the file name to generate a file ID .This step is to ensure the safety of file name.
3) Keyword Trapdoor Generator (word, Key): In this step we generate the keyword trapdoor by using word and the key which step 1 generates. Note that this trapdoor is used for encrypting word. And the function we use here is the pseudo-random f1(x).
4) Index Trapdoor Generator (Keyword Trapdoor, Fid): We generate index trapdoor by using file identifier and keyword trapdoor which generate in step 3. Note that this trapdoor is used for bloom filter index. And the function we use here is he pseudo-random f2(x) which is different from f1(x).
5) Insert Trapdoor (Index Trapdoor, Bloom filter Index): Insert trapdoor of Index into bloom filter index.
6) Build Bloom filter Index (Fid, Bloom filter Index): Build the Bloom filter Index for each file.

3.3. The Way to use Safe Index

After finishing building the safe index for each file user uploads, users can type in mutiple keywords to search files. When typing in keywords, the client use the fuzzy method, which is introduced in [8] to build fuzzy keyword set. Then do the steps 1 to 3 in building algorithm with these fuzzy keywords to generate keyword trapdoor. When server receive those trapdoors, it use the Fid and trapdoors to generates index-trapdoors and use these trapdoors to search in the Bloom-Filter Index to determine whether the keyword belongs to the file.

4. Detailed Procedures and System Security

The fuzzy keyword search runs as follows:

1) To build the safe index data owner first generates the fuzzy keyword set by using the fuzzy method called Wildcard-based fuzzy which is introduced in [8]. Then use keywords inthese sets to construct KeywordTrapdoor.After that combine it with Fid to generate Index Trapdoor. And then build the safe index with the use of Bloomfilter. When all these steps finished encrypted files are uploaded to storage server while the safe index and Fid is sent to the search server .
2) To search with keywords and key, the authorized user builds a query request by repeating steps of generating fuzzy sets and constructing KeywordTrapdoors.Then send the request to search server.
3) After receiving the query request,the cloud server uses Keyword Trapdoors and Fid to generate IndexTrapdoor. And search this trapdoor in safe index.
4) When receiving the result, user choose one of the list and send the retrieve request to the server.
5) Cloud server use the request information to search the encrypted file content, and sent it to user.
6) User decypted the file content by using his own key.

To guarantee the privacy of keywords, we use the trapdoors generated by trapdoor to create the index. And this method is proved to ensure the privacy by [4].

As for building safe index, consider two similar keywords can be deduced by comparing the over laps of 1s in the Bloom filter. Here we use two different pseudo-random function to build the Index Trapdoor. Inserting these trapdoors instead of the original keyword or the Keyword Trapdoor can avoid being attacked when we update the safe index.

5. Experiments

As far as we know, the work of [8] is most similar to ours, to evaluate the applicability we analyze the space efficiency and search efficiency between our scheme and the common inverted index(CII), which is also used to construct index in [8] rather than using bloom filter. Figure 3 shows the comparison between our fuzzy keyword search (FKS) and common inverted index (CII). With the same coefficient (as Table 2) which is a variable can control the size of fuzzy set, we can see that even building safe index by using bloom filter needs more operations, the cost of generating index for FKS and CII is still nearly the same. That means in the step for building index FKS does not create too much time cost. And in the meantime FKS has higher space efficiency.

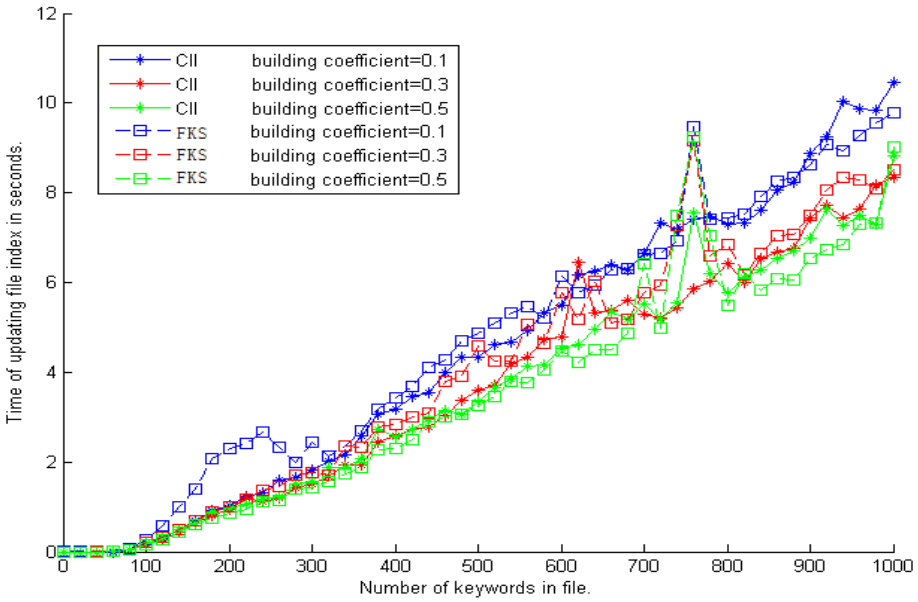

Figure 3. Time cost of generating index

The possible reason is the fuzzy keyword set and index in CII doesn't satisfy the space efficiency enough. The space it needs will be bigger with the number of fuzzy keywords or the predefined keywords for the files increases. We use the Bloom Filter to build the index. And the length of our filters is the same and never change. That means no matter how many keywords we predefined or how many fuzzy keywords we generated, the size of the filter we build for each file keeps the same.

Table 2. Parameters in experiments

Experiment variable	Meaning
Building coefficient	The variable which controls the size of fuzzy keywords set in generating index step
Searching coefficient	The variable which controls the size of fuzzy keywords set in searching files step

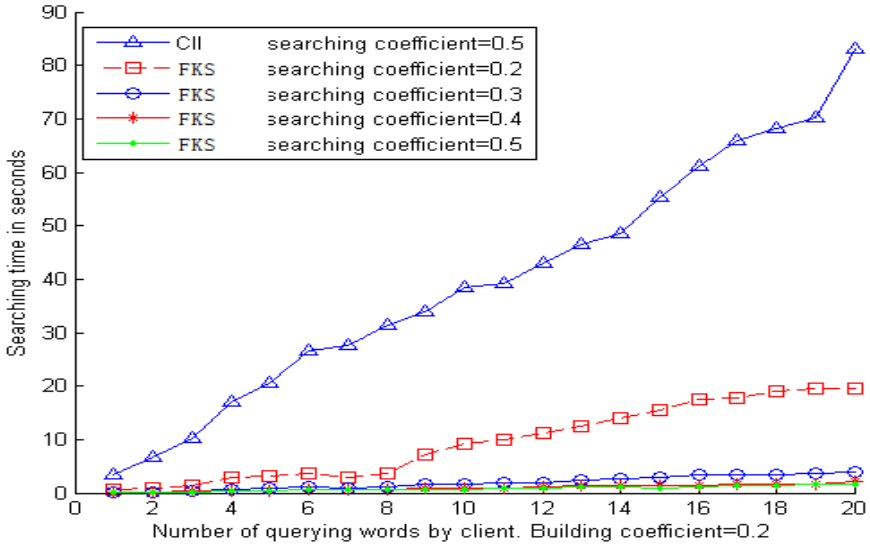

Figure 4. Comparison of searching speed with the same search keyword

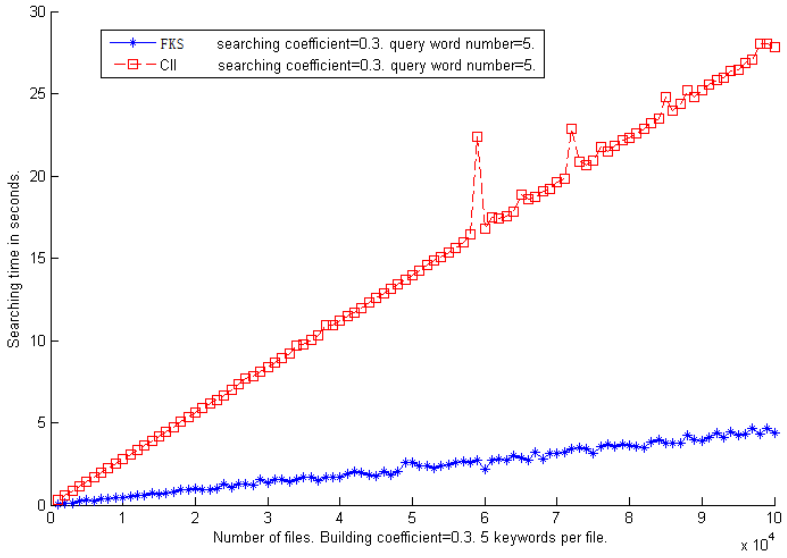

Figure 5. Comparison of searching speed with the same query word and coefficient

In Figure 4 we compare the search speed with same query keywords between FKS and CII. In our scheme, the smaller coefficient, the bigger fuzzy keyword set will be. We set the value of coefficient from 0.2 to 0.5 in FKS and 0.5 in CII. That means the fuzzy keyword set of MFKRS is bigger than CII. Figure 4shows even the fuzzy keyword set of FKS is much bigger, it is still faster than CII. In FKS, to search the keywords we use hash function, it is one of the features of Bloom Filter. Figure 5 shows the comparison of searching speed with the same query words and coefficient, obviously. The growth rate of CII is higher than FKS. It means that as the number of files increases the searching time which CII needs will be much more than FKS needs. For this aspect FKS is better than CII.

6. Conclusion

In this paper, some advanced techniques (such as Wildcard-based fuzzy set and Bloom Filter) are used to generate a storage-efficient and safe index. Based on the constructed safe index, we proposed an efficient fuzzy keyword search scheme. Experiment results show our proposed solution has high efficiency.

In our future work, we will explore other keyword semantics search (e.g., conjunction query) on encrypted data, integrity check of rank order in search result and privacy guarantees in stronger threat model. And we will try to reduce the size of fuzzy keyword set without losing the fuzzy result.

References

[1] Hong Sun, Shi-ping Chen, Li-ping Xu, Ying-ying Chen. A New Distributed Application Server System Based on Cloud Computing. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2012; 10(7): 1800-1807.
[2] Chunxia Leng, Huiqun Yu, Jingming Wang, Jianhua Huang. Securing Personal Health Records in the Cloud by Enforcing Sticky Policies. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(4).
[3] D Song, D Wagner, and A Perrig. *Practical techniques for searches on encrypted data*. In Proceedings of Security and Privacy. 2000: 44 - 55.
[4] EJ Goh. Secure indexes, Cryptology ePrint Archive, Available at URL:http://eprint.iacr.org/ 2003/216, 2003.
[5] YC Chang and M Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. Lecture Notes in Computer Science, Applied Cryptography and Network Security. 2005; 3531: 391-421.
[6] R Curtmola, JA Garay, S Kamara, and R Ostrovsky. *Searchable symmetric encryption: improved definitions and efficient constructions*. In Proceedings of ACM CCS, 2006.
[7] J Byun, D Lee and J Lim. "Efficient Conjunctive Key-word Search on Encrypted Data Storage System". Lecture Notes in Computer Science, Public Key Infrastructure. 2006; 4043: 184-196.
[8] J Li, Q Wang, C Wang, N Cao, K Ren, and W Lou. *Fuzzy keyword search over encrypted data in cloud computing*. In Proceedings of IEEE INFOCOM'10 Mini-Conference, San Diego, CA, USA; 2010.
[9] N Cao, C Wang, M Li, K Ren, and W Lou. *Privacy-Preserving Multi-keyword Ranked Search over Encrypted Cloud Data, INFOCOM*. Proceedings IEEE. 2011; 829 – 837.
[10] Q Liu, GJ Wang and J Wu. *An Efficient Privacy Preserving Keyword Search Scheme in Cloud Computing*. International Conference on Computational Science and Engineering, Vancouver. 2009; 715-720.