Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 5, May -2017

Secure and dynamic Multi-keyword Ranked Search Over Cloud Data

[1]Katre Sharad, [2]Hipparkar Pradip, [3]Sagare Ravi, [4]Hiwale Sumit

[1,2,3,4]D. Y. Patil College of Engineering, Ambi, Pune, Maharashtra, India

Abstract — *A Secure and Dynamic Multi -keyword Search the increasing popularity of cloud computing, more and more data owners. They are motivated to outsource their data to cloud servers for great convenience and reduced cost in data management. The sensitive data encrypted before outsourcing for privacy requirements. Which obsoletes data utilization like keyword-based document retrieval. Here a secure multi-keyword ranked search scheme over encrypted cloud data is presented. Which simultaneously supports dynamic update operations like that deletion and insertion of documents. The vector space model used to TFIDF model are combined, for index construction and query generation. This construct a special tree-based index structure and propose a "GDFS" algorithm is used to efficient multi-keyword search. The secure KNN algorithm is utilized to encrypt the index and query vectors. Also ensure accurate relevance score calculation between encrypted index and query vectors. To resist statistical attacks, phantom terms are added to the index vector for blinding search results. The use of our special tree-based index structure, for achieve sub-linear search time and deal with the deletion and insertion of documents with flexibility.*

General Terms

Searchable Encryption, Multi-Keyword Ranked Search, Dynamic

Update, Cloud Computing.

Keywords

CSP-Cloud Service Providers, GDFS-Greedy Depth First Search,

KNN-k-Nearest Neighbours algorithm state the efficiency of search scheme.

I. INTRODUCTION

The Increasing Popularity of use of cloud computing, data owners are aware to outsource their sensitive and complex data management system from local sites to commercial public cloud savings. For protecting the privacy of data, the sensitive data must have to be encrypted before uploading or saving on the cloud. Most of the current systems are works on plain text keyword search. The use of plain text can decrease the privacy of data, so the encrypted cloud data search is the most important than plain text keyword search over cloud data. But considering the large number of data owners, documents and data users in the cloud, it is necessary to allow multiple keyword search requests and in response returns of documents in order of their importance of keyword search. In this paper for the first time defining and solving the challenging problems of a secure and dynamic multi-key search over encrypted cloud data in cloud computing and at the same time it supports dynamic update operations like deletion and insertion of documents. The proposed scheme can achieve sub-linear search time with the deletion and insertion of documents flexibly.

i. Problem Statement

Design and implement an efficient and flexible Secure and dynamic multi-key ranked search scheme over encrypted cloud data using Greedy DFS Algorithm to provide an efficient multi-key ranked search. Use the secure KNN algorithm to encrypt the index and query vectors.

ii. Proposed System

A Secure and Dynamic Multi-keyword Search over Encrypted

Cloud Data constructed a special tree-based index structure and used a "Greedy DFS" algorithm for delivery of efficient multi-keyword ranked search. The proposed system can realize sub-linear search time and deal with the deletion and insertion of documents flexibly. Wide -ranging experiments are shown to demonstrate the efficiency of the proposed scheme.

- Plentiful works have been proposed under different risky models to accomplish various search functionality,
- Newly, some dynamic schemes have been proposed to support inserting and deleting operations on text collection.

This paper proposes a secure tree-based search system over the encrypted cloud data, which supports multi keyword search and dynamic operation on the document collection.

iii. Scope of Project

Given system built a special tree-based index structure and suggest a "GDFS" algorithm to offer effective multi-keyword search. The proposed system can realize sub-linear search time and deal with the deletion and insertion of documents openly. Extensive experiments are conducted to demonstrate the efficiency of the proposed system**.**

iv. Notations and Preliminaries

- W – The dictionary, namely, the set of keywords, denoted as W

= {w1, w2,..., wn}.

- n– The total number of keywords in W.
- Wq – The subset of W, signifying the keywords in the query.
- F – The plaintext document collection, denoted as a collection of n documents F = {f1, f2... fm}. Each document f in the collection can be considered as a sequence of keywords.
- m – The total number of documents in F.
- C – The encrypted document collection kept in the cloud server, represented as C = {c1, c2,..., cm}.
- T – The unencrypted form of index tree for the whole document collection F.
- I – The searchable encrypted tree index generated from T .
- Q – The query vector for keyword set Wq.
- td – The encrypted form of Q, which is called as trapdoor for the search request.
- Du – The index vector deposited in tree node u whose dimension matches to the cardinality of the dictionary W. Note that the node u could be also a leaf node or an internal node of the tree.

I_u – The encrypted form of D_u.

v. Vector Space Model and Relevance Score Function

Vector space model beside with tf×idf rule is broadly recycled in plaintext information recovery, which efficiently supports multi-keyword search. Here, the term frequency (tf) is the number of times a specified term (keyword) looks within a document, and the inverse document frequency (idf) is realized by dividing the cardinality of document collection by the number of documents covering the keyword. In the vector space model, every document is represented by a vector, whose elements are the normalized 'tf' values of keywords in this document. Every query is also denoted as a vector Q, whose elements are the normalized 'idf' values of query keywords in the document collection. Naturally, the distances of both the 'tf' vector and the 'idf' vector are equivalent to the entire number of keywords and the dot product of the 'tf' vector Du and the 'idf' vector Q can be calculated to quantify the relevance between the query and corresponding document. Below the notations are presented which are used in our relevance evaluation function:

- $N_{f,wi}$ – The number of keyword w_i in document f.
- N – The total no. of documents.
- N_{wi} – The no. of documents that contain keyword w_i.
- $tf'_{f,wi}$ – The tf value of w_i in document f.
- idf'_{wi} – The idf value of w_i in document collection.
- $tf_{u,wi}$ – The normalized 'tf' value of keyword w_i stowed in index vector D_u.

- idf_{wi} – The normalized 'idf' value of keyword w_i in document collection.

The relevance evaluation function is defined as:

RScore

$$(D_u, Q) = D_u \cdot Q = \sum_{w_i \in W_q} TF_{u,w_i} \times IDF_{w_i} \quad \ldots. \quad (1)$$

If u is an internal node of the tree, $tf_{u,wi}$ is calculated from index Vectors in the child nodes of u. If the u is a leaf node, $tf_{u,wi}$ is calculated as:

tfu,wi = tf'f,wi √∑wi W(tf'f,wi)2 (2)

And in the search vector Q, idfwi is calculated as:

Where tf'f,wi = 1+lnNf,wi. ∈

idfwi = idf'wi √∑wi Wq(idf'wi)2 (3)

Where idf'wi = ln(1 + N/Nwi).

Keyword Balanced Binary Tree. The balanced binary tree is broadly used to deal with optimization problems. The keyword balanced binary (KBB) tree in our system is a dynamic data structure whose node stores a vector D. The elements of vector D are the normalized TF values. Sometimes, we refer the vector D in the node u to Du for simplicity. Formally, the node u in our KBB⟨ tree is defined ⟩as follows:

u = ID,D,Pl,Pr,FID , (4)

Where, ID denotes the identity of node u, Pl and Pr are respectively the pointers to the left and right child of node u. If the node u is a leaf node of the tree, FID supplies the identity of a document, and D indicates a vector consisting of the normalized tf values of the keywords to the document. If the node u is an internal node, FID is set to null, and D denotes a vector containing the tf values which is calculated as follows:

D[i]= max{u.Pl → D[i],u.Pr → D[i]},i = 1,...,m. (5)

II. IMPLEMENTATION

In secure and dynamic multi-keyword rank search scheme on cloud, Data owner up-loads encrypted file in cloud. Data owner uploads many file in cloud. As cloud is large storage and can contains many file so it must have to efficient for user to select or search files. Therefore, index is created for file and that encrypted index is uploaded on cloud with encrypted file.

Data owner or data user make changes in file or can perform file operations on file can insert or delete data so data or files require by user must be updated .So index of files are automatically updated after changes made on file. As index is in hierarchical format nodes of file are change according to changes in file.

In addition, trapdoor is generated for user to search by using multiple keywords of files. This trapdoor search for file matching keywords and then matching files are send to users.

i. Methodology Used

Algorithm 1

AES Algorithm The encryption process uses a set of specially derived keys called round keys. These applied, along with other operations, on an array of data that holds exactly one block of data the data to be encrypted. This array we call the state array.

You take the following AES steps of encryption for a 128-bit block:

1. Derive the set of round keys from the cipher key.
2. Initialize the state array with the block data (plaintext).
3. Add the initial round key to the starting state array.
4. Perform nine rounds of state manipulation.
5. Perform the tenth and final round of state manipulation.
6. Copy the final state array out as the encrypted data (cipher text).

The reason that the rounds have been listed as "nine followed by a final tenth round" is because the tenth round involves a slightly different manipulation from the others. These algorithm are used to file content are convert plaint text to cipher text.
Note: AES is a non-Festal cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds. The key size, which can be 128, 192, or 256 bits, depends on the number of rounds.

Algorithm 2

MD5 algorithm description MD5 algorithm takes input message of arbitrary length and generates 128-bit long output hash. MD5 hash algorithm consist of five steps:

Step 1. Append Padding Bits

Step 2. Append Length

Step 3. Initialize MD Buffer

Step 4. Process Message in 16-Word Blocks

Step 5. Output

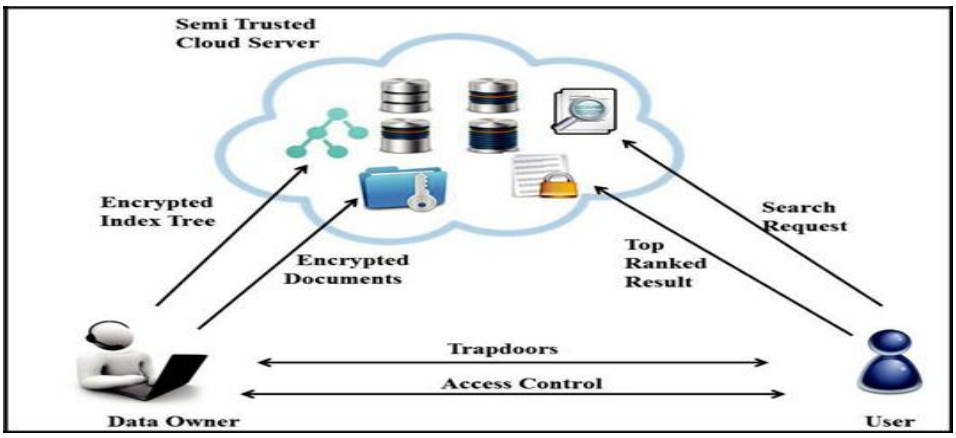

Fig. 1: System Architecture

III. RESULT ANALYSIS

File Size	File Upload	File Download
10(KB)	0.05	0.03
50(KB)	1.75	1.77
100(KB)	2.5	2.6
200(KB)	4.8	4.9

Table 1: Result analysis of system

After testing this system gives the result as shown in above table. *The* above table shows the time efficiency at different file size for example, to upload a file of 10KB require 0.05 min and to download it require 0.03 min.

The bar diagram of the result is shown below.

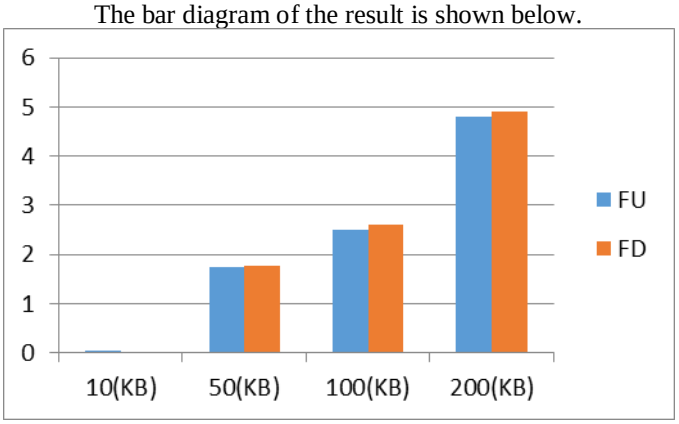

Chart 1: Bar diagram of result analysis

IV. CONCLUSIONS

A protected, efficient and element inquiry plan is proposed, which bolsters the exact multi-essential word positioned hunt as well as the dynamic erasure and insertion of archives. We build an extraordinary decisive word adjusted parallel tree as the record, and propose a" Greedy Depth-first Search" calculation to acquire preferred efficiency over straight pursuit. In addition, the parallel inquiry procedure can completed for further diminish the time cost. The plan's securities are ensured against two danger models by utilizing the protected kNN calculation. Trial results show the efficiency of our proposed plan. There are still numerous test issues in symmetric SE plans. In the proposed plan, the information proprietor is in charge of producing overhauling data and sending them to the cloud server. Therefore, the information proprietor needs to store the decoded record tree and the data that are important to recalculate the IDF values. Such a dynamic information proprietor may not be exceptionally suitable for the distributed computing model. It could be a significant however; difficult future work to plan an element searchable encryption plot whose overhauling operation can be finished by cloud server just, in the interim holding the capacity to bolster multi-pivotal word positioned pursuit. What's more, as the large portion of works about searchable encryption, our plan essentially considers the test from the cloud server. Really, there are numerous safe difficulties in a multi-client plan. Firstly, every one of the clients more often than not keep the same secure key for trapdoor era in a symmetric SE plan. For this situation, the client's disavowal is error.

ACKNOWLEDGEMENT

In performing our assignment, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. The completion of this assignment gives us much Pleasure. We would like to show our gratitude Prof. Smita Bhosale for giving us a good guideline for this research throughout numerous consultations. We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in writing this research.

REFERENCES

[1] K. Ren, C.Wang, Q.Wang et al.,"Security challenges for the public cloud," IEEE Internet Computing, Vol. 16, No. 1, 69–73, 2012.

[2] S. Kamara, K. Lauter,"Cryptographic cloud storage," In Financial Cryptography and Data Security. Springer, 2010, 136– 149.

[3] C. Gentry,"A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, G. Persiano, "Public key encryption with keyword search," In Advances in Cryptology- Eurocrypt 2004. Springer, 2004, 506–522.

[5] D. Boneh, E. Kushilevitz, R. Ostrovsky, W. E. Skeith III,"Public key encryption that allows pir queries," In Advances in Cryptology-CRYPTO 2007. Springer, 2007, 50–67.

[6] D. X. Song, D. Wagner, A. Perrig,"Practical techniques for searches on encrypted data," In Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on. IEEE, 2000,44–55.

[7] Y.-C. Chang, M. Mitzenmacher,"Privacy preserving keyword searches on remote encrypted data," In Proceedings of the Third international conference on Applied Cryptography and Network Security. Springer-Verlag, 2005, 442–455.

[8] R. Curtmola, J. Garay, S. Kamara, R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," In Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006,79–88.

BIOGRAPHIES

Katre Sharad, D. Y. Patil College of Engineering, Ambi, Pune, Maharashtra, India.

Hipparkar Pradip, D. Y. Patil College of Engineering, Ambi, Pune, Maharashtra, India.

Sagare Ravi, D. Y. Patil College of Engineering, Ambi, Pune, Maharashtra, India.

Hiwale Sumit, D. Y. Patil College of Engineering, Ambi, Pune, Maharashtra, India.

