International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol. 8, Issue.12, December 2020 E-ISSN: 2347-2693

Privacy Preserving Search Over Encrypted Data with Secure and Dynamic Operation in Cloud Computing

Poonam Patil[1*], Seema Mane[2]

[1,2]Computer Engineering, Computer Engineering, GOVT. Residence women Polytechnic, Tasgaon, India

*Corresponding Author: poonam28july@gmail.com

DOI: https://doi.org/10.26438/ijcse/v8i12.3338 | Available online at: www.ijcseonline.org

Received: 09/Oct/2020, Accepted: 10/Dec/2020, Published: 31/Dec/2020

***Abstract*—** Due to the increasing significance of cloud computing, additional and tremendous information of data owners are made to outsource their information to cloud servers for excellent convenience and to lessen cost in information management. However, sensitive information ought to be encrypted before outsourcing for privacy needs that obsoletes information utilization like keyword-based document retrieval. In this paper, a secure multi-keyword search method over encrypted cloud data is described. At the same time it also supports dynamic update operations like deletion and insertion of documents. For index construction and query generation, the vector space model and widely-used TF_IDF model both are combined. A special tree- based index structure is constructed and a "Greedy Depth-first Search" method is proposed to produce economical multi-keyword ranked search. The secure KNN is used to encrypt the index and vectors. It also guarantees about correctness in appropriate score calculation between encrypted index and query vectors. In order to stop statistical attacks, phantom terms are intercalary to the index vector for accurate search results. As special tree-based index structure is used, the proposed method is able to do sub- linear search time and also handle the deletion and insertion of documents flexibly.

***Keywords*—** Searchable encryption, multi-keyword ranked search, dynamic update, cloud computing.

I. INTRODUCTION

Allotted computing has been considered as one more model of huge trade IT base, which can sort out monstrous asset of processing, stockpiling and functions, and empower clients to appreciate pervasive, high quality and on-curiosity approach entry to an original pool of configurable registering assets with uncommon efficiency and negligible economic overhead. Pulled in by means of these enticing add-ons, each humans and undertakings are spurred to outsource their expertise to the cloud, alternatively than acquire programming and apparatus to take care of the know-how themselves. Regardless of the exclusive facets of curiosity of cloud administrations, outsourcing sensitive knowledge, (for example, messages, character wellbeing documents, organization finance know- how, government documents, and so on.) to faraway servers brings protection considerations. The cloud govt supplier (CSPs) that keep the understanding for clients could get to purchasers' moody information without approval. A general option to take care of cozy the knowledge confidentiality is to encode the know-how earlier than outsourcing. Be that as it'll, this may occasionally convey about a colossal fee as regards understanding ease of use. For request, the present systems on magical word centered data.

Constitution of cloud computing are greatly used on the plaintext information, can't be frankly useful on the encrypted data. Downloading all the knowledge from the cloud and decrypt close by means of is certainly impractical. With the intention to tackle the above challenge, researchers have designed some basic-purpose options with utterly- homomorphic encryption or ignorant RAMs. Nonetheless, these methods are usually not sensible because of their excessive computational overhead for each the cloud sever and person. On the opposite, extra practical distinctive- rationale options, comparable to searchable encryption (SE) scheme have made specific contributions in phrases of efficiency, functionality and safety. Searchable encryption schemes enable the patron to retailer the encrypted information to the cloud and execute keyword search over ciphertext area. Up to now, profuse works have been proposed below exceptional risk items to reach quite a lot of search functionality, similar to single key phrase search, similarity search, multi-keyword boolean search, ranked search, multi- keyword ranked search, and so on. With them, multi- key phrase ranked search achieves increasingly interest for its practical applicability. Just lately, some dynamic schemes had been proposed to aid inserting and deleting operations on record collection. These are significant works as it is particularly feasible that the info owners have got to replace their knowledge on the cloud server. However few of the dynamic schemes support efficient multi- key phrase ranked search.

II. LITERATURE SURVEY

A. Practical Techniques for Searches on Encrypted Data
It's desirable to store knowledge on data storage servers

such as mail servers and file servers in encrypted type to reduce security and privacy risks. However this often implies that one has to sacrifice functionality for safety. For instance, if a client desires to retrieve best documents containing certain words, it was not earlier recognized how one can let the data storage server participate in the hunt and answer the query without loss of knowledge confidentiality. On this paper, we describe our cryptographic schemes for the challenge of shopping on encrypted information and provide proofs of protection for the ensuing crypto programs. Our techniques have a number of vital benefits. They are provably at ease. They furnish provable secrecy for encryption, in the experience that the untrusted server cannot gain knowledge of something in regards to the plaintext when simplest given the cipher text; they provide query isolation for searches, which means that the untrusted server cannot learn something extra about the plaintext than the search effect; they furnish managed looking, so that the untrusted server cannot seek for an arbitrary phrase without the consumer's authorization; they also support hidden queries, in order that the person could ask the untrusted server to seek for a secret phrase without revealing the word to the server. The algorithms we gift are easy, fast (for a document of size n, the encryption and search algorithms most effective need O(n) flow cipher and block cipher operations), and introduce nearly no house and communication overhead, and for that reason are realistic to use today.

B. Public Key Encryption That Allows PIR Queries

Consider the subsequent main issue: Alice needs to carry here mail utilizing a storage-supplier Bob (such as a Yahoo! Or hot mail electronic message account). This storage-supplier ought to furnish for Alice the potential to gather, retrieve, search and delete emails however, even as, ought to learn neither the content material of messages dispatched from the senders to Alice (with Bob as Associate in Nursing middleman), nor the search standards utilized by Alice. A trivial resolution is that messages can doubtless be dispatched to Bob in encrypted kind and Alice, whenever she needs to hunt for a few message, can raise Bob to ship her replica of the whole info of encrypted emails. This withal is hugely inefficient. We are going to be fascinated with choices that are account economical and, while, respect the privacy of Alice. During this paper, we have a tendency to show the way to produce a public-key secret writing theme for Alice that allows PIR browsing over encrypted files. Our answer is that the initial to disclose no partial experience regarding the consumer's search (including the entry sample) within the public-key surroundings and with non-trivially tiny communication complexness. This provides a theoretical choice to a state of affairs posed by method of Boneh, DiCrescenzo, Ostrovsky and Persiano on "Public-key secret writing with keyword Search." The principal technique of our answer in addition makes it doable Single-Database PIR writing with sub-linear account complexness, that we have a tendency to bear in mind of freelance interest.

C. Public Key Encryption with keyword Search

We study the matter of looking out on knowledge that's encrypted employing a public key system. Take into account user Bob WHO sends email to user Alice encrypted below Alice's public key. Associate in Nursing email entryway desires to check whether or not the e-mail contains the keyword \urgent" so it might route the e-mail consequently. Alice, on the opposite hand doesn't would like to present the entryway the flexibility to decipher all her messages. We have a tendency to outline and construct a mechanism that allows Alice to produce a key to the entryway that allows the entryway to check whether or not the word \urgent" could be a keyword within the email while not learning the rest concerning the e-mail. we have a tendency to visit this mechanism as PublicKey coding with keyword Search. As another example, take into account a mail server that stores varied messages publically encrypted for Alice by others. Mistreatment our mechanism Alice will send the mail server a key that may modify the server to spot all messages containing some special keyword, however learn nothing else. We have a tendency to outline the idea of public key coding with keyword search and provide many constructions.

D. A Fully Homomorphic Encryption Scheme Author:

We propose the first absolutely homomorphic coding theme, resolution a central open downside in cryptography. Such a theme permits one to figure whimsical functions over encrypted knowledge while not the secret writing key – i.e., given encryptions E(m1),...,E(mt) of M1,...,mt, one will efficiently figure a compact ciphertext that encrypts f(m1,...,mt) for any efficiently calculable operate f. This downside was exhibit by Rivest et al. in 1978. Absolutely homomorphic coding has various applications. as an example, it allows non-public queries to a look engine – the user submits associate degree encrypted question and therefore the program computes a compendious encrypted answer while not ever staring at the question within the clear. It conjointly allows looking on encrypted knowledge – a user stores encrypted files on a distant file server and may later have the server retrieve solely files that (when decrypted) satisfy some Boolean constraint, even if the server cannot decode the files on its own. A lot of broadly speaking, absolutely homomorphic coding improves the efficiency of secure multiparty computation.

III. THE SYSTEM MODEL

A dynamic searchable coding theme, whose change operation will be completed by cloud server solely, is the meantime reserving power to support multi-keyword hierarchal search. If it's required to revoke a user during this theme, we want to reconstruct the index and distribute the new secure keys to all or any the licensed users. The system model in this paper involves three different entities: data owner, data user and cloud server, as illustrated in Fig. 1.

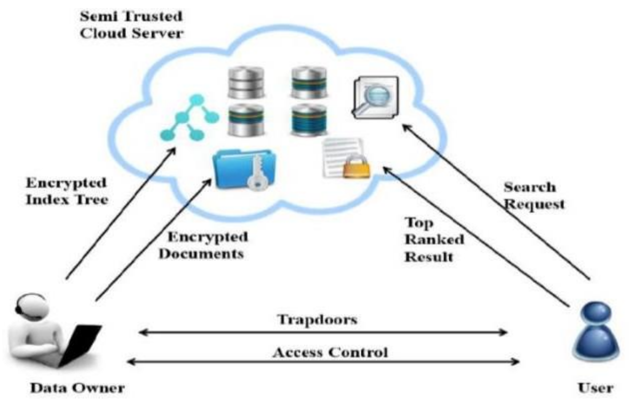

Figure 1 System Architecture

The system model during this paper incorporates 3 clear substances: information homeowners, information user and cloud server, as illustrated in Figure one. There are unit multiple information owner in system. As information owner incorporates a gathering of records F = that he has to source to the cloud server in encoded structure whereas up 'til currently keeping the flexibility to ascertain on them for convincing utilization. information owner first manufactures a secure searchable tree index I from archive accumulation F, and a brief time later makes an encrypted document gathering C for F. a short span later, the info owner outsources the encoded accumulation C and therefore the secure index I to the cloud server, and safely disseminates the key information of trapdoor era and document decoding to the approved information users. In addition, the info owner is aware of his documents hold on within the cloud server. Whereas, change the information owner creates the upgrade information domestically and sends it to the server can also perform data dynamic operations on files. Data user's area unit approved ones to induce to the archives of knowledge owner. With question keywords, the approved user will produce a trapdoor TD as indicated by search management mechanisms to induce k encrypted documents from cloud server. By then, decode the documents with the shared secret key. Cloud server stores the encrypted document accumulation C and therefore the encrypted searchable tree index I for information owner. Within the wake of tolerating the trapdoor TD from the info user, look over the index tree I, finally offers back the relating gathering of top-k settled encoded reports. Also, within the wake of tolerating the update data from the info owner, the server has to update the index I and document gathering C as per the received data. After insertion or deletion of a record, we have a tendency to need change synchronously the index. Since the index of DMRS theme is planned as a balanced binary tree, the dynamic operation is finished by redesigning hubs within the list tree. The report on record is simply visible of archive acknowledges, and no entrance to the substance of records is needed

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

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

IV. PRIVACY PRESERVING TECHNIQUES

Privacy-Preserving: The system is designed to preserve privacy of cloud server from getting extra information about documents, index tree and query. The main privacy requirements are stated as:

1. *Index Confidentiality and Query Confidentiality*: the plain text information, TF values that are stored in the index tree and IDF values of query keywords etc. all should be prevented from cloud server.
2. *Trapdoor Unlinkability:* the cloud server should not come to know about whether two queries (trapdoors) that are encrypted are generated from same search request.
3. *Keyword Privacy:* By analyzing the statistical available information, cloud server could not recognize the specific keyword. Here, unencrypted dynamic multi-keyword ranked search (UDMRS) scheme is described. This is constructed using vector space model and KBB tree.

A. *Index Construction of UDMRS Scheme*

KBB index tree structure helped in index generation. For each document there is need to generate tree node. These nodes are termed as leaf nodes of the index tree. On the basis of these leaf nodes internal nodes are formed. An example of our index tree is shown in Figure the data structure of the tree node is defined as ⟨*ID;D; Pl; Pr; FID*⟩, where the unique identity *ID* for each tree node is generated through the function GenID(). • *CurrentNodeSet*– The set of current processing nodes which have no parents. If the number of nodes is even, the cardinality of the set is denoted as $2h(h \in \mathbf{Z}+)$, else the cardinality is denoted as $(2h + 1)$. • *TempNodeSet*– The set of the newly generated nodes.

B. *Search Process of UDMRS Scheme*

The search process of the UDMRS scheme uses recursive procedure. This procedure follows "Greedy Depth first Search (GDFS)" algorithm. The result list RList is constructed and that element is defined as RScore which is named as relevance score calculated as per formula (1).The k accessed documents are stored in RList and these have largest relevance scores. As per RScore, the elements are arranged in descending order. These elements are updated time to time while search process.

RScore(Du;Q) – The function to calculate the relevance score for query vector Q and index vector Du stored in node u, which is defined in Formula (1).

- k^{th}*score*– This is smallest relevance score in RList. This is initialized to 0.
- *hchild*– This indicates child node of tree node having greater relevance score.

C. BDMRS Scheme

With the help of KNN algorithm, we make the basic dynamic multi-keyword ranked search (BDMRS) scheme. In this scheme the goal is to achieve privacy in cipher text model. The following are the steps described:

- SK ← Setup() 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,2) two (m×m) invertible matrices M1 and M2. Namely, SK = {S;M1;M2}.
- I ← GenIndex(F; SK) First, the unencrypted index tree T is built on F by using T ← BuildIndexTree(F). Secondly, the data owner generates two random vectors {Du Finally, the encrypted index tree I is built where the node u stores two encrypted index vectorsIu
- TD ← GenTrapdoor(Wq; SK) using Wq, the unencrypted query vector Q is generated. This vector has length m. If wi∈Wq, Q[i] stores the normalized IDF value of wi; else Q[i] is set to 0. The two random vectors are formed by splitting the query vector Q.The difference is that if S[i] = 0, Q′[i] and Q′′[i] are set to two random values whose sum equals to Q[i]; else Q′[i] and Q′′[i] are set as the same as Q[i]. Finally, the algorithm returns the trapdoor TD = {M −1 1 Q′;M −1 2 Q′′}.
- RelevanceScore ← SRScore(Iu;TD) Relevance score of node u is calculated by cloud server. This is done by using trapdoor TD and in index tree I.

D. EDMRS Scheme

The Index Confidentiality and Query Confidentiality in the known cipher text model is protected by BDMRS scheme. But, by tracking the path of already visited nodes, cloud server can connect search requests. The cloud server can identify a keyword as normalized TF distribution. This can be obtained from relevance scores. The calculated relevance score of Iu and TD is same as Du and Q. To improve security, one of the methods is to break such similarity. In order to change or disturb the relevance score calculation, there is need to introduce some randomness. In addition, to suit different users' preferences for higher accurate ranked results or better protected keyword privacy, the randomness are set adjustable. The enhanced EDMRS scheme is almost the same as BDMRS scheme except that: • SK ← Setup() In this algorithm, we set the secret vector S as a m-bit vector, and set M1 and M2 are (m + m′) × (m + m′) invertible matrices, where m′ is the number of phantom terms. • I ← GenIndex(F; SK) Before encrypting the index vector Du, we extend the vector Du to be a (m+m′)-dimensional vector. Each extended element Du[m+ j], j = 1; :::;m′, is set as a random number "j . • TD ← GenTrapdoor(Wq; SK) The query vector Q is extended to be a (m + m′)-dimensional vector. A number of m′′ elements are chosen randomly and set as 1, and the others are set to 0. • *RelevanceScore* ← SRScore(*Iu;TD*) After the execution of relevance evaluation by cloud server, the final relevance score for index vector *Iu*equals to *Du* · *Q* + Σ′′*v*, where *v* ∈*{j*|*Q*[*m* + *j*] = 1*}*.

E. Dynamic Update Operation of DMRS

The dynamic operation is performed by updating nodes in the index tree as the index of DMRS scheme is balanced binary tree. Note that the update on index is merely based on document identifies, and no access to the content of documents is required *{I′s; ci}* ← GenUpdateInfo(*SK; Ts; i; updtype*)). The updated information {Is; ci} made by this algorithm will be sent to cloud server. If *updtype* is equal to *Del*, the data owner deletes from the subtree the leaf node that stores the document identity *I* and updates the vector *D* of other nodes in subtree *Ts*, so as to generate the updated subtree *T ′* If *updtype*is equal to *Ins*, the data owner generates a tree node *u* = ⟨GenID()*;D; null; null; i*⟩ for the document *fi*, where *D*[*j*] = *TFfi;wj*for*j* = 1*; :::;m*. Then, the data owner inserts this new node into the subtree *Ts*as a leaf node and updates the vector *D* of other nodes in subtree*Ts {I′; C′}* ← Update(*I; C; updtype; I′s; ci*) In this algorithm,cloud server replaces the corresponding subtree *Is*(the encrypted form of *Ts*) with *I′ s*, so as to generate a new index tree *I′*. If *updtype*is = *Ins,* cloud server inserts the encrypted document *ci* into *C*, obtaining a new collection *C′*. If *updtype*is = *Del*, cloud server deletes the encrypted document *ci* from *C* to obtain the new collection *C′*.

V. ALGORITHM

Step1: Initialization -The data owner randomly generates the secret key K = (S;M1;M2) Where, S is a (m+1)-dimensional binary vector. -M1 and M2 are two (m + 1) × (m + 1) invertible matrices. -Data owner sends (K; sk) to search users through a secure channel, sk-SECRET KEY.

Step 2: Index Building - The data owner firstly utilizes symmetric encryption algorithm (e.g., AES) to encrypt the document collection (*F*1*; F*2*; · · · ; FN*) with the symmetric key *sk.* **-**Ecrypted Document is *Cj*(*j* = 1*;* 2*; · · · ;N*). - Data owner generates an *m*-dimensional binary vector *P* according to *Cj*(*j* = 1*;* 2*; · · · ;N*), where each bit *P*[*i*] indicates whether the encrypted document contains the keyword *wi*, i.e., *P*[*i*] = 1 indicates yes and *P*[*i*] = 0 indicates no. Then she extends *P* to a (*m* + 1)-dimensional vector *P′*, where *P′*[*m* + 1] = 1. - If *S*[*i*] = 0(*i*= 1*;* 2*; · · · ;m* + 1), *pa*[*i*] and *pb*[*i*] are both set as *P′*[*i*]; If *S*[*i*] = 1(*i*= 1*;* 2*; · · · ;m* + 1), the value of *P′*[*i*] will be randomly split into *pa*[*i*] and *pb*[*i*] (*P′*[*i*] = *pa*[*i*]+*pb*[*i*]). - Then, the index of encrypted document *Cj*can be calculated as *Ij*= (*paM*1*; pbM*2). -Data owner send *Cj* ||*FIDj* ||*Ij*(*j* = 1*;* 2*; · · · ;N*) to cloud.

Step 3: Trapdoor Generation - The search user firstly generates the keyword set f*W*for searching. Then, she creates a *m*-dimensional binary vector *Q* according to f*W*, i.e., *Q*[*i*] = 1 indicates yes and Q[*i*] = 0 indicates no - The search user extends *Q* to a (*m* + 1)-dimensional vector *Q′*, where *Q′*[*m* + 1] = −*s* - The search user chooses a random number r >0 to generate *Q′′* = *r* · *Q′*. Then she splits *Q′′* into two (*m* + 1) vectors (*qa; qb*): if *S*[*i*] = 0(*i*= 1*;* 2*; · · · ;m*+ 1), the value of *Q′′*[*i*] will be randomly split into *qa*[*i*] and $qb[i]$. - $S[i] = 1 (i = 1; 2; \cdots ; m + 1)$, $qa[i]$ and $qb[i]$ are both set as $Q''[i]$.

Step4: Query - With the index $I_j (j = 1; 2; \cdots ; N)$ an, the cloud server calculates the query result as:

$$\begin{aligned} R_j = I_j \cdot T_{\widetilde{W}} &= (p_a M_1, p_b M_2) \cdot (M_1^{-1} q_a, M_2^{-1} q_b) \\ &= p_a \cdot q_a + p_b \cdot q_b = P' \cdot Q'' \\ &= r P' \cdot Q' = r \cdot (P \cdot Q - s) \end{aligned}$$

VI. PREPARE RESULTS AND DISCUSSION

Precision and Privacy: The search precision of scheme is affected by the dummy keywords in EDMRS scheme. Here, the 'precision' is defined]: Pk= k′=k, where k′ is the number of real top-k documents in the retrieved k documents.

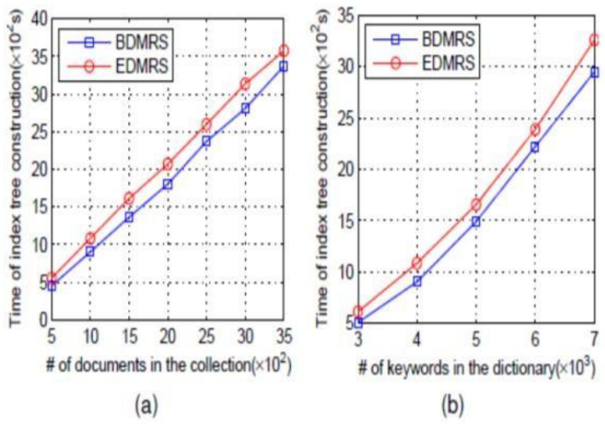

Fig. 2 Time cost for trapdoor generation: (a) for different sizes of dictionary with the fixed number of query keywords $t = 10$, and (b) for different numbers of query keywords with the fixed dictionary,$m= 1000$

In the EDMRS scheme, phantom expressions are added to index vector. Due to this the calculation of relevance score becomes unclear. Hence the cloud server cannot identify the keywords even by using TF distributions. Here, we measure the ambiguity of the relevance score using "rank privacy". This is defined as:

$$P'k = \Sigma |ri - r'i| = k2;$$

Where,
ri = the rank number of document in the retrieved top-k documents
$r'I$ = its real rank number in the whole ranked results.
If the rank is larger, the security of the scheme is higher.

Efficiency

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 by dividing operation and doing multiplications of $m * m$ matrix. The index tree is formed by using post order traversal method. This will be done based upon document collection F. The number of nodes generated during traversal is $O(n)$.

The time taken for each node in the process is as follows: index vector generation = $O(m)$, for vector splitting $O(m)$ and for multiplications of matrix $O(m2)$. As a whole, the time complexity for index tree construction is $O(nm2)$. The cost of time for constructing index tree depends upon the number of elements in document collection F and number of keywords in dictionary W.

2. Trapdoor Generation

To generate trapdoor, there is need to perform vector splitting operation and do multiplications of (m * m) matrix. For this the time complexity is $O(m^2)$.

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, while doing real search the lots of nodes are not accessed. We denote the number of leaf nodes that contain one or more keywords in the query. Probably, the numbers of leaf nodes are larger than the number of required documents k, but they are far less than the number of elements of the document collection n. The height of the index of tree is retained as log n because it is a balanced binary tree. The complexity of relevance score calculation is $O(m)$.

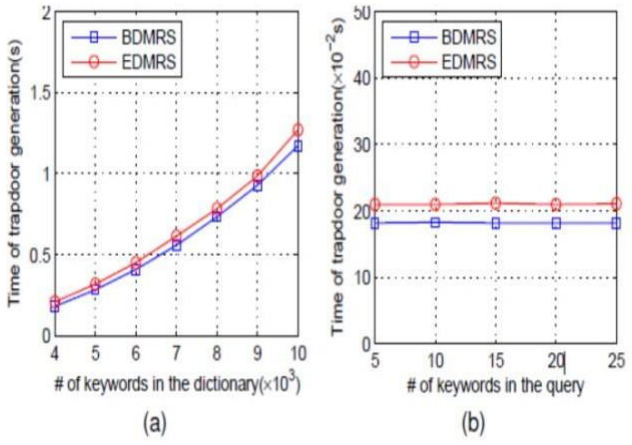

Fig. 3 Time cost for trapdoor generation: (a) for different sizes of dictionary with the fixed number of query keywords, $t = 10$, and (b) for different numbers of query keywords with the fixed dictionary, $m= 4000$.

4. Update Efficiency

To update a leaf node, there is need to update log n nodes. At each node, the encryption operation is to be performed on index vector. It takes $O(m^2)$ time. Also the time complexity of update operation is $O(m2)$.

VII. CONCLUSION

A secure, economical and dynamic search theme is planned, that supports not solely the correct multi-keyword stratified search however additionally the dynamic deletion and insertion of documents. We construct a special keyword balanced binary tree because the index, and propose a "Greedy Depth-first Search" formula to get higher potency than linear search. In addition, the parallel search method may be administered to more cut back the time price. The security of the theme is protected against 2 threat models by victimization the secure KNN formula. Experimental results demonstrate the potency of our planned theme. There are still several challenge issues in parallel SE schemes. In the planned theme, data owner is accountable for generating change information and causation them to the cloud server. Thus, data owner must store the unencrypted index tree and therefore the information that are necessary to cipher the force values. Such an energetic information owner might not be terribly appropriate for the cloud computing model. It may well be a significant however troublesome future work to style a dynamic searchable coding theme whose change operation may be completed by cloud server solely, meantime reserving the ability to support multi-keyword ranked search. In addition, because the most of works regarding searchable coding, our theme principally considers the challenge from the cloud server. Actually, there are several secure challenges in a very multi-user theme. Firstly, all the users sometimes keep identical secure key for trapdoor generation in a very parallel SE theme. In this case, the revocation of the user is huge challenge. If it's required to revoke a user during this theme, we want to construct the index and distribute the new secure keys to any or all the approved users. Secondly, parallel SE schemes sometimes assume that each one the info users square measure trustworthy. It is not sensible and a dishonest information user can cause several secure issues. For example, a dishonest information user might search the documents and distribute the decrypted documents to the unauthorized ones. Even more, a dishonest information user might distribute his/her secure keys to the unauthorized ones. In the future works, we are going to try and improve the SE theme to handle these challenge issues.

REFERENCES

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

[2]. B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multikeyword fuzzy search over encrypted data in the cloud," in *IEEE INFOCOM*, **2014**.

[3]. S. Kamara and K. Lauter, "Cryptographic cloud storage," in Financial Cryptography and Data Security. Springer, **pp. 136–149. 2010.**

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

[5]. E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," in *Proceedings of the 6th Theory of Cryptography Conference on Theory of Cryptography*. Springer-Verlag, **pp. 457–473, 2009.**

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

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

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

[9]. B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multikeyword fuzzy search over encrypted data in the cloud," in *IEEE INFOCOM*, **2014**.

[10].Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," in *Proceedings of the First international conference on Pairing-Based Cryptography*. Springer-Verlag, **pp. 2–22, 2007.**

[11].D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proceedings of the 4th conference on Theory of cryptography*. Springer-Verlag, **pp. 535–554, 2007.**

[12].B. Zhang and F. Zhang, "An efficient public key encryption with conjunctive-subset keywords search," *Journal of Network and Computer Applications*, **vol. 34, no. 1, pp. 262–267, 2011.**

[13].J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *Advances in Cryptology–EUROCRYPT 2008*. Springer, **pp. 146–162, 2008.**

[14].E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," in *Proceedings of the 6th Theory of Cryptography Conference on Theory of Cryptography*. Springer-Verlag, **pp. 457–473, 2009.**

[15].A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption," in *Proceedings of the 29th Annual international conference on Theory and Applications of Cryptographic Techniques*. Springer-Verlag, **pp. 62–91, 2010.**

[16].A. Swaminathan, Y. Mao, G.-M. Su, H. Gou, A. L. Varna, S. He, M.Wu, and D.W. Oard, "Confidentiality-reserving rank-ordered search," in *Proceedings of the 2007 ACM workshop on Storage security and survivability*. ACM, **pp. 7–12, 2007**.