International Journal of Scientific Research in Science and Technology
Print ISSN: 2395-6011 | Online ISSN: 2395-602X (www.ijsrst.com)
doi : https://doi.org/10.32628/IJSRST

A Privacy-Preserved & Encrypted Multi-Keyword Ranked Search in Cloud Storage

Ms. Radhika S Landge[1]*, Prof. Nitin R. Chopde[2]
[1] M.E Scholar, Department of Computer Science & Engineering, G. H. Raisoni College of Engineering & Management, Amravati, Maharashtra, India
[2]Assistant Professor, Department of Computer Science & Engineering, G. H. Raisoni College of Engineering & Management, Amravati, Maharashtra, India

Article Info
Volume 8, Issue 2
Page Number : 387-396

Publication Issue
March-April-2021

Article History
Accepted : 12 April 2021
Published : 15 April 2021

ABSTRACT

Recently, the growth of private and semi-private data has accelerated on the data network, and instruments to track such data have exploded in security safeguarding. In the field of data systems, security saving seeking is becoming increasingly important in order to conduct various information mining operations on encoded data stored in various stockpiling frameworks. It is also a critical and difficult task to maintain the confidentiality of private information exchanged among specialist co-ops and data owners. One possible structure provided by the existing system is defense safeguarding ordering (PPI). In this framework, archives are stored on a private server in plain content format in exchange for secrecy. To make this framework more safe and reliable, we first store the records on the server in scrambled form, and then use the Key Distribution Center (KDC) to allow decoding of information obtained from the private server at the customer's end. To improve the client's look engagement, we also use TF-IDF, which provides efficient results positioning. Finally, we run a series of large tests on the dataset to evaluate how well our proposed system works. The proposed system would be shown to be superior to any current one in terms of security safeguarding, proficient and stable inquiry on scrambled appropriated archives based on exploratory findings.

Keywords : Cloud computing, Encryption, Inner product similarity, Single Keyword Search, Multi-keyword search, ranking.

I. INTRODUCTION

In today's world, a large number of people are using the internet. Constantly new data is outsourced as a result of development away from consumer needs, and is then effectively semi-placed in servers. Cloud enlisting is a Web-based model, where cloud clients can supply their data into the cloud [1]. Through storing data in the cloud, data owners are able to remain unconstrained once the cap has been reached. As a result, ensuring the authenticity of confidential data is an important task. When all is said and done, when it comes to data protection in the cloud, the data owner must be outsourced in an encoded structure to individuals, and the data process must be based on plaintext keyword look. The capable measure of "orchestrate organizing" is chosen. To measure the parallel whole, sort out organizing is used. Encourage the centrality of data records to the request address keywords by encouraging them to be organized. The request office, as well as the security guarding mixed cloud data, are critical. If we concentrate a large number of data reports and data customers in the cloud, it will be difficult to meet the demands of execution, convenience, and adaptability. The enormous amount of data files in the cloud server fulfills to come about imperative rank instead of returning indistinguishable outcomes, stressing to experience the genuine data recovery. To recover the request correctness, the locating arrangement considers a large number of keyword interests. In today's Google organize look for gadgets, data customers give a game plan of keywords rather than a noteworthy keyword look centrality to recover the most outrageous fundamental data. Compose preparation is the synchronized coordination of query keywords that are essential for the response to the request. It remains the captivating work for how to relate the mixed cloud looks for as a result of inherence prosperity and stability. The difficult problem of multi-keyword located search over encoded cloud data is solved by using strict security requirements followed by various multi-keyword semantics. We choose promote preparation from a variety of multi-keyword based semantics. Our roles are as follows: 1) For the first case, we examine the issue of multi keyword located mixed cloud data and create a game plan of stringent security requirements for such a secure cloud data utilization system. 2) In two specific hazard models, we suggest two MRSE arrangements based on the similarity measure of "organize preparation" while also following various assurance requirements. 3) A thorough analysis into the proposed arrangements' protection and profitability confirmations is provided; a review of this current reality dataset also reveals that the proposed plots in fact introduce low overhead on count and correspondence.

II. LITERATURE SURVEY

Secure and confirmation saving keyword search was proposed by Qin Liu et al. in [1]. By using open key encryption, it provides keyword authentication, information confirmation, and semantic security. The rule problem with this pursuit is that encryption and unscrambling have a higher correspondence and computational cost.

Authorized Private Keyword Search (APKS) is a concept proposed by Ming Li et al. in [2.] Keyword protection, index and query privacy, fine-grained search authorization and revocation, multi-dimensional keyword search, scalability, and efficiency are all provided. This intrigue scheme increases the efficacy of the pursuit by using a consistency chain of noteworthiness, but after a short period of time, each of the characteristics has been leveled.

Cong Wang et al. proposed Secure and Efficient Ranked Keyword Search in [3], which highlights prepare overhead, information and keyword security, and the least communication and figuring overhead. It isn't useful for various keyword missions, and it comes with a small amount of overhead in the record-keeping process.

Secured padded keyword search for with symmetric searchable encryption was proposed by Kui Ren et al. [4]. (SSE). It can't play out different keywords semantic pursue with open key based searchable encryption, and it can't strengthen delicate excitement with open key based searchable encryption. Upgrades to the padded searchable report are not completed competently.

Ming Li et al. [5] proposed a searchable disseminated stockpiling framework with privacy guarantees. SSE (Scalar-Product-Preserving Encryption) and Order-Preserving Symmetric Encryption are used to perform the encryption. It strengthens the fundamentals of security and utilitarianism. The open key based searchable encryption is not strengthened by this strategy.

K-gram-based fluffy keyword Ranked Search was suggested by Wei Zhou et al. [6]. In this case, the proprietor creates a k-gram delicate keyword appeal for record D, and the tuple I, D> is traded to the demand server (SS), which is embedded in the development channel for size power. To the point of confinement server, the blended record D is exchanged. Regardless, the problem is that the padded keyword set is determined by the k-measure gram's in relation to the jacquard coefficient.

The Secure Channel Free Public Key Encryption with Keyword Search (SCF-PEKS) system was proposed by J. Baek et al. in [7]. In these device package servers, each server creates its own unique open and private key pair, but KGA attacks this approach from the outside.

Trapdoor in recognisability Public-Key Encryption with Keyword Search was introduced by H. S. Rhee et al. [8]. (IND-PEKS). This outsourcing is done in the form of SCF-PEKS. It encounters an external aggressor who uses KGA to isolate the rehash of the keyword trapdoor case.

Peng Xu et al. [9] suggested PEFKS with Fuzzy Keyword Search, in which the client creates padded keyword trapdoor Tw for W and right keyword trapdoor Kw for W. Tw is requested by the client for CS. After that, CS tests Tw with a delicate keyword record and sends a superset of sorting out figure messages using CS's Fuzz Test estimation. Exact Test is a client process that involves verifying that the figure works with Kw and recovering the encoded data. For large databases, the process of creating padded keyword archives and performing a keyword check is difficult.

Ning et al. [10] suggested Privacy Preserving Multi Keyword Ranked Search as a solution to this issue (MRSE). It's useful for content models with established figures and foundations that display blended data. It has a low overhead in terms of computations and correspondence. For multi-keyword searches, the workplace arrangement is selected. The disadvantage is that MRSE has a very low standard deviation, which reduces keyword security.

III. PROBLEM STATEMENT

The expansive number of information clients and archives in cloud, it is essential for the hunt administration to permit multi-keyword question and give result likeness positioning to meet the compelling information recovery require. The searchable encryption concentrates on single keyword inquiry or Boolean keyword look, and once in a while separates the list items.

a) Single-keyword search without Ranking
b) Boolean-keyword look without Ranking
c) Single-keyword hunt with Ranking

We define and address the testing problem of security saving multi-keyword positioned look over scrambled cloud data (MRSE), as well as provide a set of stringent security requirements for such a secure cloud data use system to become a reality. We choose the proficient rule of "facilitate organizing" from among the various multi-keyword semantics. Over scrambled cloud data, a multi-keyword search is performed (MRSE). Internal object comparability is used to "coordinate coordinating."

IV. PROPOSED SOLUTION

We propose a framework under which any endorsed customer may conduct an interest search on mixed data using various keywords without disclosing the keywords he is looking for or the data of the records that fit the query. Confirmed customers may use the cloud to search for structures using unmistakable keywords in order to recover the correlated reports. Our recommendation system enables a group of customers to request the database if they have specified trapdoors for the hunt words that enable customers to consolidate them in their request. Our proposed system will perform several keyword searches in a single query and place the results such that the customer only receives the most important matches. Similarly, we devise a set of strict security requirements. We choose the feasible control of "sort out preparation" from among various multi-keyword semantics.

V. SYSTEM OVERVIEW

The system architecture is stressed by making a direct assistant structure for a structure. It portrays the general edge of the wander which rapidly delineates the working of the structure and the inspiration driving the wander stage is to mastermind an answer of the issue recognized by the need archive. The underneath Figure 1 exhibit the system of the structure. We consider three segments in our structure designing: Data Owner, Data customer and Cloud Server.

- Data Owner is in charge of the making of the database.
- Data Users are the devotees in a gathering who can utilize the documents of the database.
- Cloud Server bargains information offices to confirmed clients. It is fundamental that server be torpid to substance of the database it keeps.

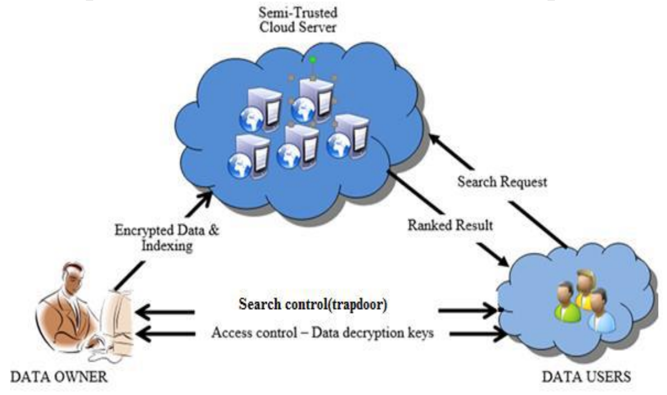

Semi-Trusted Cloud Server
Search Request
Ranked Result
Encrypted Data & Indexing
Search control(trapdoor)
Access control – Data decryption keys
DATA OWNER
DATA USERS

Figure 1: Search over Encrypted Cloud

The data owner has a large number of data records that he wants to outsource to a cloud server in a hybrid environment. Prior to outsourcing, the data owner would first create a secured searchable record using a series of keywords omitted from the report creation process, and then store both the list and the encoded archive on a cloud server. We make every effort to complete the underwriting between the data owner and the customers. A guaranteed customer makes and shows a request in a hidden casing a trapdoor of the keyword to the cloud server to search the record gathering for a given keyword. Following the receipt of a chase order, the server is responsible for searching for the record and returning the customer's report preparing course of action. We believe that the ensured situated keyword is dangerous because it requires the following: the query yield must be retrieved in accordance with simple situated noteworthiness rules, ensuring that customers' records are recovered precisely. In any case, cloud servers must inspect any information that is dark or irrelevant about the fundamental rules, as they expose vital sensitive data against keyword insurance. To slow down the exchange, the customer can send as many as k possible considerations near the trapdoor, and the cloud server will only send back the top-k most relevant archives to the customer's concerned keyword. Plot Objectives: To allow for localized performance for professional use of outsourced cloud data under the already defined display, our device setup can quickly complete security and execution attestations as follows.

Multi-keyword Ranked Search: To design look for arrangements which allow multi-keyword question and give result closeness situating to fruitful data recuperation, instead of returning undifferentiated results.

Insurance Preserving: To shield the cloud server from taking in additional data from the dataset and the record, and to meet security.

Viability: Expert with low correspondence and estimation overhead should be used to achieve the above goals on helpfulness and protection. Mastermind Matching: "Compose preparation" [2] is a common likeness measure that assesses the importance of a chronicle to the request by counting the number of query keywords that appear in the response. When customers identify the correct subset of the dataset to be retrieved, Boolean requests are completed successfully with the customer's defined chase requirement. Customers are more adaptable to interpret a list of keywords displaying their tension and recoup the most critical reports with a rank query.

VI. METHODOLOGY

A. Stemming:

Stemming is a method for reducing twisted (or sometimes determined) words to their pledge stem, base, or root shape—for the most part, a composed word frame—in phonetic morphology and data recovery. The stem may not have to be indistinguishable from the word's morphological foundation; it is usually sufficient that related words lead to a common stem, even though this stem is not a significant root in and of itself. Since the 1960s, stemming calculations have been considered in software engineering. Many web indexes treat terms that have indistinguishable origins from equivalent words as a query extension, a process known as conflation. The terms stemming calculations and stemmers are often used to describe stemming projects.

A stemmer for English, for example, should identify the string "cats" (and possibly "catlike", "catty" etc.) as based on the root "cat", and "stems", "stemmer", "stemming", "stemmed" as based on "stem". A stemming algorithm reduces the words "fishing", "fished", and "fisher" to the root word, "fish". On the other hand, "argue", "argued", "argues", "arguing", and "argus" reduce to the stem "argu" (illustrating the case where the stem is not itself a word or root) but "argument" and "arguments" reduce to the stem "argument".

B. Suffix-stripping algorithms:
Suffix-stripping algorithms don't depend on a query table that comprises of curved structures and root frame relations. Rather, a commonly littler rundown of "tenets" is put away which gives a way to the calculation, given an information word shape, to discover its root frame. A few cases of the principles include:

- if the word ends in 'ed', remove the 'ed'
- if the word ends in 'ing', remove the 'ing'
- if the word ends in 'ly', remove the 'ly'

When the maintainer is sufficiently trained in the difficulties of etymology and morphology, as well as encoding postfix stripping laws, addition stripping approaches have the advantage of being significantly easier to keep up with than savage constrain calculations. Because of the poor execution when handling remarkable ties (like "ran" and "run"), addition stripping calculations are sometimes regarded as crude. With a few exceptions, the arrangements provided by postfix stripping calculations are limited to those lexical groups that have clearly understood additions. Nonetheless, this is a problem since not all aspects of debate have such a well-thought-out set of criteria. The aim of lemmatization is to improve this test.

C. Stop-Words:
Stop words may be words that are sifted through before or after usual dialect information is handled in the registration process (text). Despite the fact that stop words often refer to the most commonly known words in a dialect, there is no single all-inclusive list of stop words used by all common dialect preparation apparatuses, and in fact, not all devices even use such a list. To support state seek, a few apparatuses specifically avoid evaporating these stop terms.

For any purpose, any grouping of terms can be chosen as the stop words. These are the most popular, short ability terms for some web crawlers, such as the, is, at, which, and on. Stop words, particularly in names like "The Who," "The," and "Take That," can cause problems when scanning for phrases that contain them in this situation. Other web crawlers remove the most common terms from a query, including lexical words like "need," in order to improve execution.

One of the pioneers of data recovery, Hans Peter Luhn, is credited with coining the phrase and putting it into practice. The words "stop word," "stop rundown," and "stoplist," which do not appear in Luhn's 1959 introduction, appear in the writing in the blink of an eye a short time later.

A forerunner concept was used in the development of a few concordances. For example, the main Hebrew concordance, Meir local, had a one-page list of unindexed terms, with no meaningful relational words and conjunctions that are similar to stop words today.

D. TF-IDF
TF-IDF stands for term recurrence in contrast to archive recurrence, and the TF-IDF weight is a common weight used in data recovery and content mining. This weight is a factual metric for determining how important a word is to a record in a collection or corpus. The meaning of a word grows in proportion to the number of times it appears in the archive, but is balanced by the word's recurrence in the corpus. Online search tools often use variations of the TF-IDF weighting plan as a focal apparatus in scoring and positioning an archive's value in response to a client inquiry.

Summing the TF-IDF for each query term yields one of the simplest positioning capacities; various more complex positioning capacities are variants of this simple model.

In a variety of subject areas, such as material outline and characterization, TF-IDF may be used to effectively separate stop-words.

The tf-idf weight is usually made up of two terms: the primary processes and the standardized Word Frequency (TF), also known as. The number of times a word appears in a text, divided by the total number of words in that archive; the second term is the Inverse Document Frequency (IDF), calculated as the logarithm of the number of documents in the corpus divided by the number of records where the term appears.

TF: Term Frequency is a metric that calculates how much a term appears in a study. Since each document is unique in length, it's possible that a word will appear in much more circumstances in longer records than in shorter ones. In this vein, as a form of standardization, the term recurrence is periodically divided by the report duration (also known as the total number of terms in the record):

TF (t) = (Number of times term t appears in a document) / (Total number of terms in the document).

IDF: Inverse Document Frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scale up the rare ones, by computing the following:

IDF (t) = log_e (Total number of documents / Number of documents with term t in it)

E. Build Index Tree

Input: the document collection F = {f1, f2, ..., fn} with the identifiers F_{ID} = {F_{ID}—F_{ID} = 1, 2... n }.

Output: the index tree T

```
1.  for each document {$F_I$} in F do
2.  Construct a leaf node u for fFID,
3.  Insert u to CurrentNodeSet;
4.  end for
5.  while the number of nodes in CurrentNodeSet is larger than 1 do
6.  if the number of nodes in CurrentNodeSet is even, i.e. 2h then
7.  for each pair of nodes u0 and u00 in CurrentNodeSet do
8.  Generate a parent node u for u0 and u00 ,
9.  Insert u to TempNodeSet;
10. end for
11. else
12. for each pair of nodes u0 and u00 of the former (2x2) nodes in CurrentNodeSet do
13. Generate a parent node u for u0 and u00 ;
14. Insert u to TempNodeSet;
15. end for
16. Create a parent node u1 for the (2h - 1)-th and 2h-th node, and then create a parent node u for u1 and the (2h + 1)-th node;
17. Insert u to TempNodeSet;
18. end if
19. Replace CurrentNodeSet with TempNodeSet and then clear TempNodeSet;
20. end while
21. return the only node left in CurrentNodeSet, namely, the root of index tree T ;
```

F. BDMRS

SK $\longleftarrow$ 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, and 2) two (m X m) invertible matrices M1 and M2. Namely, SK = {S, M1, M2}.

I $\longleftarrow$ GenIndex (F, SK) First, the unencrypted index tree T is built on F by using

T$\longleftarrow$ BuildIndexTree (F) Secondly, the data owner generates two random vectors (D'u, D"u) for index vector Du in each node u, according to the secret vector S. Specifically, if S[i] = 0, D'u[i] and D"u[i] will be set equal to Du[i]; if S[i] = 1, D'u[i] and Du"u[i] will be $\{M_1^T D'_u, M_2^T D''_u\}$ set as two random values whose sum equals to Du[i]. Finally, the encrypted index tree I is built where the node u stores two encrypted index vectors Iu=

TD $\longleftarrow$ GenTrapdoor (Wq, SK) with keyword set Wq, the unencrypted query vector

Q with length of m is generated. If wi "Wq, Q[i] stores the normalized IDF value of wi; else Q[i] is set to 0. Similarly, the query vector Q is split into two random vectors Q' and 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 $\{M_1^{-1} D'_u, M_2^{-1} D''_u\}$ same as Q[i]. Finally, the algorithm returns the trapdoor TD =

Relevance Score $\longleftarrow$ SRScore (Iu, TD) With the trapdoor TD, the cloud server computes the relevance score of node u in the index tree I to the query.

G. EDMRS Scheme

The enhanced EDMRS scheme is almost the same as BDMRS scheme except that:

SK$\longleftarrow$ 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 $\longleftarrow$ 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 $\longleftarrow$GenTrapdoor (Wq, SK) The query vector Q is extended to be a (m + m')- dimensional vector. Among the extended elements, a number of m" elements are randomly chosen to set as 1, and the rest are set as 0.

Relevance Score $\longleftarrow$ SRScore(Iu, TD) After the execution of relevance evaluation by cloud server, the final relevance score for index vector Iu equals to Du ^A $\Sigma \varepsilon v$, where v ε $\{j|Q[m+j] = 1\}$

I. Implementation

A. Data User Module:

Clients of this framework are information clients, who have their identities set up to retrieve documents from the cloud that are exchanged by information proprietors. Since the number of documents stored on the cloud server could be large, the client is accommodated in an intrigue office. On the cloud server, the client should be able to perform a multi-keyword search. Once the result for a specific interest is available, clients should be able to use the system to send a request to the individual details owners of the document (also known as a trap-section request) for the documents to be downloaded. Clients of information will be given a demand support screen that will show whether the information proprietor has perceived or rejected the demand. Clients should be able to download the decoded record if the request has been approved.

B. Information Owner Module:

The owners of the data should be able to share records in this module. Before transferring the documents to the cloud, the reports are encrypted. The data owners are also given the option of entering keywords for the records that are sent to the server. These keywords are used for the requesting purpose, which aids in the interest return values being returned quickly. When these documents have been uploaded to the cloud, consumers should be able to search for them using keywords. The data owners will also be provided with a request approval screen, allowing them to approve or deny the requests received by the data customers.

C. Document Upload and Encryption Module:

The owners of the data should be able to share archives in this module. Before sending the files to the cloud, the documents are combined. The data owners are given a different choice when it comes to entering the keywords for the records that are sent to the server. These keywords are used for the requesting purpose, which aids the chase's ability to quickly return values. When these documents are opened in the cloud, consumers should be able to search for them using keywords. The data proprietors will also be outfitted with a request underwriting screen, allowing them to approve or deny requests received from data customers. Before being exchanged, the record should be encrypted with a key, so that data customers cannot simply download it without it. The data customers will request this key via the trap-portal. Unapproved consumers would not be able to download these archives since they are encrypted using RSA figuring.

D. Document Download and Decryption Module:

Data clients are clients on this framework that will have their identities ready to retrieve documents from the cloud that the information proprietors have transferred. Since the amount of records stored on the cloud server may be enormous, the client is given a pursuit office. On the cloud server, the client should be able to perform a multi-keyword search. When the result for a specific search appears, clients should be able to submit a request to the individual information owners of the document via the process (also known as a trap-entryway request) to download these documents. The information clients will also be presented with a demand endorsement screen, which will indicate whether the information owner has accepted or rejected the request. Clients should be able to download the unscrambled document if the request has been approved. The record should be unscrambled with a key before being downloaded. The details clients will inquire about this key through the trap-entryway request. The information clients will be able to download and use the record once the key is provided during the download.

E. Rank-Search Module:

This module enables the information clients to search for the reports with multi-keyword rank looking. This model uses the on occasion utilized rank pursuing figuring down present the yield for multi-keywords. "Energize Matching" administer will be gotten a handle on for the multi-keyword pursuing. This module in like way oversees making an archive for speedier pursue.

VII. EXPERIMENTAL RESULT

Fig. 2 shows look time correlation diagram; in roar chart X-hub demonstrates the calculation by which records are sought while Y-pivot indicate time required for seeking question related in ms.

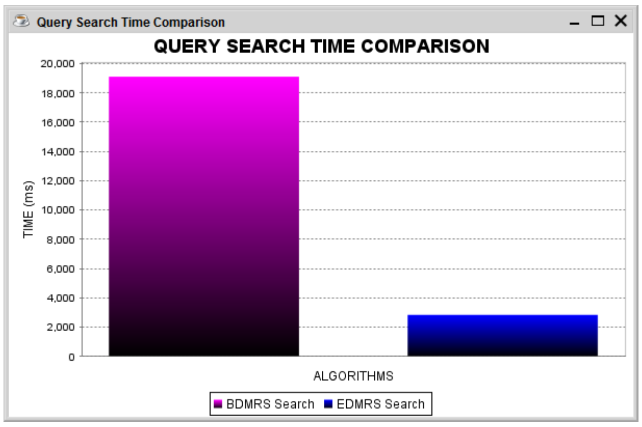
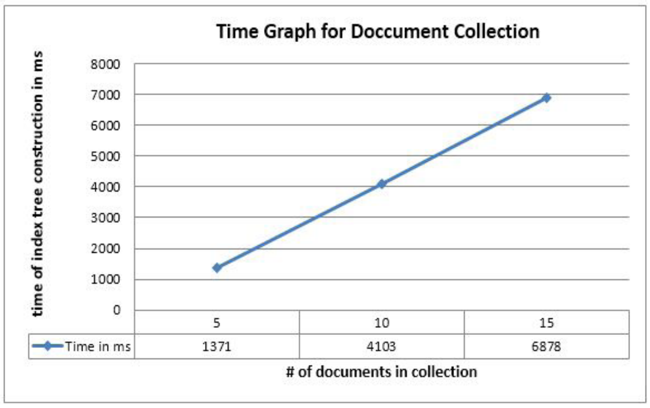


Figure 2: Query Search Time Comparison

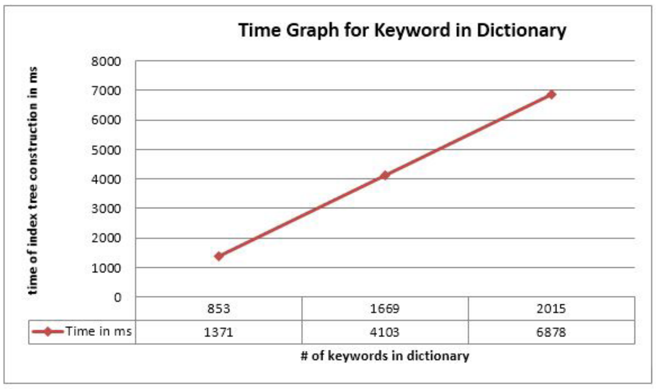
Fig. 3 shows time diagram; in above chart X-pivot indicates number of records in gathering while Y-hub demonstrate time required for producing file tree in ms, with increment in number of archives the time required to create list tree is additionally increment.



	5	10	15
Time in ms	1371	4103	6878

Figure 3: Time Graph for Document Collection

Fig. 4 shows time diagram; in above chart X-hub indicates number of keywords in word reference while Y-pivot demonstrate time required for producing file tree in ms, with increment in number of keywords the time required to create list tree is additionally increment.



	853	1669	2015
Time in ms	1371	4103	6878

Figure 4: Time Graph for Keyword in Dictionary

VIII. CONCLUSION

First, we illustrate and resolve the challenging of multi-keyword positioned look over scrambled cloud data, as well as create a set of security requirements in this job. We choose the persuasive resemblance measure of "facilitate coordinating," i.e., as many different matches as possible, from among various multi-keyword semantics to accurately capture the relevance of outsourced archives to the query correspondence. In the future, we'll focus on promoting other multi-keyword semantics over encoded data and verifying the rank request's honesty in the item keywords. We suggest an important thought of MRSE for the traditional test of consistent multi-keyword semantics without security breaks. After that, we present two improved MRSE diagrams to account for a variety of stringent security requirements in two different risk models. A detailed review of the proposed plans' security and effectiveness guarantees is given, and tests on this current reality data set show that our future frameworks have low overhead on both measurement and correspondence.

IX. REFERENCES

[1]. Qin Liuy, Guojun Wangyz, and Jie Wuz,"Secure and privacy preserving keyword searching for cloud storage services", ELSEVIER Journal of Network and computer Applications, March 2011

[2]. Ming Li et al.," Authorized Private Keyword Search over Encrypted Data in Cloud Computing,IEEE proc. International conference on distributed computing systems, June 2011,pages 383-392

[3]. Cong Wang et al.,"Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data", IEEE Transactions on parallel and distributed systems, vol. 23, no. 8, August 2012

[4]. Kui Ren et al., "Towards Secure and Effective Data utilization in Public Cloud", IEEE Transactions on Network, volume 26, Issue 6, November / December 2012

[5]. Ming Li et al.,"Toward Privacy-Assured and Searchable Cloud Data Storage Services", IEEE Transactions on Network, volume 27, Issue 4, July/August 2013

[6]. Wei Zhou et al., "K-Gram Based Fuzzy Keyword Search over Encrypted Cloud Computing "Journal of Software Engineering and Applications, Scientific Research , Issue 6, Volume 29-32,January2013

[7]. J. Baek et al., "Public key encryption with keyword search revisited", in ICCSA 2008, vol. 5072 of Lecture Notes in Computer Science, pp. 1249 - 1259, Perugia, Italy, 2008. Springer Berlin/Heidelberg.

[8]. S. Rhee et al., "Trapdoor security in a searchable public-key encryption scheme with a designated tester," The Journal of Systems and Software, vol. 83, no. 5, pp. 763-771, 2010.

[9]. Peng Xu et al., Public-Key Encryption with Fuzzy Keyword Search: A Provably Secure Scheme under Keyword Guessing Attack",IEEE Transactions on computers, vol. 62, no. 11, November 2013

[10]. Ning Cao et al.," Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data", IEEE Transactions on parallel and distributed systems, vol. 25, no. 1, jan 2014

[11]. D. X. D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proc. S & P, BERKELEY, CA, 2000, pp. 44.

[12]. Wang, N. Cao, K. Ren, and W. J. Lou, Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data, IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 8, pp. 1467-1479, Aug. 2012.

[13]. W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Privacy-preserving multi-keyword text search in the cloud supporting

similarity-based ranking," in Proc. ASIACCS, Hangzhou, China, 2013, pp. 71-82.

[14]. R. X. Li, Z. Y. Xu, W. S. Kang, K. C. Yow, and C. Z. Xu, Efficient multi-keyword ranked query over encrypted data in cloud computing, Futur. Gener. Comp. Syst., vol. 30, pp. 179-190, Jan. 2014.

[15]. Gurdeep Kaur, Poonam Nandal, "Ranking Algorithm of Web Documents using Ontology", IOSR Journal of Computer Engineering (IOSR-JCE) eISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. VIII (May-Jun. 2014), PP 52-55

Cite this article as :

Radhika S Landge, Prof. Nitin R. Chopde, "A Privacy-Preserved & Encrypted Multi-Keyword Ranked Search in Cloud Storage", International Journal of Scientific Research in Science and Technology (IJSRST), Online ISSN : 2395-602X, Print ISSN : 2395-6011, Volume 8 Issue 2, pp. 387-396, March-April 2021.
Journal URL : https://ijsrst.com/IJSRST218271