EFURMS: An Efficient Scheme for File Upload and Ranked Multi-Keyword Search over Encrypted Data in Cloud

Bhavya M, Thriveni J, Venugopal K R

Abstract: ***Cloud based services provide scalable storage capacities and enormous computing capability to enterprises and individuals to support big data operations in different sectors like banking, scientific research and health care. Therefore many data owners are interested to outsource their data to cloud storage servers due to their huge advantage in data processing. However, as the banking and health records usually contain sensitive data, there are privacy concerns if the data gets leaked to un-trusted third parties in cloud storage. To protect data from leakage, the widely used technique is to encrypt the data before uploading into cloud storage servers. The traditional methods implemented by many authors consumes more time to outsource the data and searching for a document is also time consuming. Sometimes there may be chances of data leakage due to insufficient security. To resolve these issues, in the current VPSearch(VPS) scheme is implemented, which provides features like verifiability of search results and privacy preservation. With its features the current system consumes more time for file uploading and index generation, which slows down the searching process. In the existing VPS scheme time minimization to efficiently search for a particular document is a challenging task on the cloud. To resolve all the above drawbacks, we have designed an index generation scheme using a tree structure along with a search algorithm using Greedy Depth-first technique, that reduces the time for uploading files and file searching time. The newly implemented scheme minimizes the time required to form the index tree file for set of files in the document which are to be uploaded and helps in storing the files in a index tree format. These techniques result in reducing the document upload time and speeding up the process of accessing data efficiently using multi-keyword search with top-'K' value.***

Keywords: Cloud Data Security Scheme, Cloud storage, Efficiency, Ranked Multi-keyword Search and Tree Based Indexing.

I. INTRODUCTION

Cloud Computing is one of the emerging technology that has been booming in the last few years. Due to its great computing capabilities, storage, and various other applications [1], it has become a significant platform for distributed operations especially as information management and data storage service. Cloud based computing is a collection of applications and resources which are used to store or retrieve data remotely. These features greatly minimize software/hardware price and system overhead and permits the authorized transfer channels to share data resources between data users and cloud. Cloud service providers like Amazon, Microsoft Azure and other private organizations have data centres deployed in cloud storage. At any given time, the data can be added or removed by the data owner from the cloud and the user can access the data at any time. The cloud based services charge the users on a pay-per-use basis for the number of resource used. The key features of a cloud based service provider are data privacy, security and reliability. The limitations in public cloud to provide data privacy and confidentiality results in data loss. Providing confidentiality and security of information in case of a breach, to prevent leakage of data uploaded to the cloud is crucial. To solve this, the data is pre-encrypted before uploading to the cloud based storage. Data encryption is typically used for data privacy. It helps to convert plaintext into cipher text using algorithmic based schemes and mathematical models, which is in an encrypted form to un-trusted third-parties. Different types of data encryption schemes have been implemented [2], which are used to encrypt the data before uploading it to the cloud. Applying these schemes for data encryption leads to enormous costs and data processing techniques implemented for plaintext data do not work on encrypted data. The keyword search techniques designed for traditional database data retrieval cannot be used directly on the encrypted data. Hence, processing queries over encrypted data and assuring data privacy is a challenging task. There are many techniques designed to perform a search over encrypted data, for example with the single keyword search or multi-keyword search methods. Single keyword-search is not efficient for advanced queries and leads to high computation costs. Therefore multi-keyword search is used, which is practical and efficient than a single keyword search. Keyword search is the most frequent operation performed on cloud based services. Multiple researches have been done with the help of search engine optimization techniques, which are designed to achieve data privacy. There are schemes like Multi-keyword Ranked for Searchable Encryption (MRSE) [3] [4], the Homomorphic MAC encryption process and the Random Challenge process [5] [6]. MRSE is an efficient, valid and authentic way to export cloud data. The MRSE system uses the Homomorphic MAC encryption process to improve the privacy of exported data. Random challenge process is used to verify top-k search results, by evaluating and finding incorrect top-k results with high probability. In these methods, we look at the time to calculate the top indicators, the time to load data in the clouds and the search time.

Manuscript received on January 08, 2021.
Revised Manuscript received on January 15, 2021.
Manuscript published on February 28, 2021.
* Correspondence Author
Bhavya M*, Department of Computer Science and Engineering University Visvesvaraya College of Engineering Bengaluru - 560001, India Email: uvcebhavya04@gmail.com
Thriveni J, Department of Computer Science and Engineering University Visvesvaraya College of Engineering Bengaluru - 560001, India
Venugopal K R, Vice Chancellor Banglore University, Bengaluru - 560056, India

To refine these parameters, we have used the tree-based index building process [7] to reduce the calculation time and loading time. And the combination of tree-based indexing and Greedy-Depth first technique helps us to reduce search time and produce faster results. Finally, a Random Challenge process is used to validate top-k search results and collected by greedy search technique.

A. *Research Contribution*

In the newly proposed scheme, we make use of a Greedy depth first search technique and Tree-Based index generation technique for effective and multi-keywords search. The proposed scheme provides effective, efficient and secure search results in a short period of time and keeps confidential information away from unauthorized users and cloud service providers. Our contribution is summarized as follows:

- The proposed scheme enhances efficiency, reduces search time and document upload time on big databases.
- Index computation time and the document upload time is lowered using the Tree based Index generation algorithm and its exclusive features.
- The proposed system can achieve the best search results using a Greedy depth-first search algorithm.
- Protecting sensitive data and better confidentiality of keywords are achieved in the proposed system
- The new process yields better top-k search results

B. *Organization of the paper*

The organization of the paper is as follows: Section II discusses the work-related literature. Section III describes the background work, which highlights the important schemes used in the proposed system. A proposed system model and its description and function flow is explained in section IV. Section V presents the comparison between the proposed and the existing schemes. Finally, conclusions are highlighted in section VI.

II. RELATED WORK.

Cloud computing service provides massive storage capabilities to enterprises and individuals with computing power. With these facilities many more data owners are outsourcing their data on cloud storage for higher accessibility in data mining and management. Storage of sensitive data brings about privacy concerns if data is leaked to un-trusted parties in the cloud. To protect data from leakage, encrypt data before outsourcing which reduces data utility and causes data retrieval obsolete. To resolve these issues, Ding *et al.*, [8] have proposed a random traversal algorithm that yields different index paths for the same query to protect query data privacy. They designed a group multi-keyword top-k search scheme based on partition which constructs tree based indexes for all documents and improves the search efficiency. These techniques are experimentally validated and proved that the proposed scheme is more efficient and secure than the existing methods. In the existing methods, authors concentrate on single-keyword search over encrypted data on the block chain. They necessarily execute a single-keyword search multiple times and take the intersection of results to obtain a multi-keyword scheme. This causes efficiency, data privacy, long delays in the system process and extra cost in calculating the intersection of multiple sets. To resolve these issues and drawbacks, Jiang *et al.*,[9] have proposed a bloom filter-enabled multi-keyword search protocol. This protocol selects low-frequency keywords, which removes the majority of data from the results and reduces the computational cost. The pseudorandom tag implementation helps to complete search operation in a single round and privacy is preserved. The proposed methods perform 14.7 percent less time delay and 59.96 percent less financial cost. Data owners are influenced to deploy their data management systems from local sites to the public cloud storage for high flexibility and budget savings. When operating large data sets it is imperative to allow multiple keywords in the search request and return relevant documents to these keywords. In this paper, Anjali [10] has implemented a scheme to solve the problem of privacy-preserving multi-keyword ranked search (MRSE) over encrypted data in the cloud storage system by selecting the efficient similarity measures of co-ordinate matching. She has also implemented inner product similarity to tentatively estimate the similarity measures. In this proposed system author tries to achieve various privacy requirements with improved MRSE. With the advanced features of cloud storage, data owners opt to outsource their data to the cloud storage system. Privacy preservation is a major concern for sensitive data, therefore it is compulsory to encrypt the data before outsourcing it to the cloud. There are many searchable encryption methods to ensure data availability. In the case of a multi-owner scheme, the existing search systems need to pay more attention to the efficiency of data user's queries. To improve the efficiency of the search scheme, Peng *et al.*, [11] have introduced a tree-based ranked multi-keyword search scheme for multiple data owners (TBMSM). They developed a multi-keyword search scheme with top-k ranked search results using TF × IDF model, by considering large data set in the cloud. Based on the bilinear mapping, they have proposed a novel privacy-preserving search protocol for secure search. Each data owner constructs the tree-based index and encrypted it for security purposes. Finally, the cloud server merges these indexes using a depth-first search algorithm to search similar files. With these above implementation results, the authors concluded that the proposed scheme is efficient and secure. More and more industries and individuals choose cloud storage to outsource their data with the fast evolution of cloud computing services. Data is encrypted before uploading it into the cloud to provide security or preserve data privacy. It is a difficult task to perform a search over encrypted data. Dai *et al.*,[12] have proposed a privacy-preserving multi-keyword ranked search scheme over encrypted data in a hybrid cloud called MRSE-HC. For index search, the keyword partition-based bit vectors are used for documents and queries. With the MRSE-HC scheme, authors have proposed an enhanced MRSE-HC scheme called EMRSE-HC to improve the search efficiency. These schemes improve search efficiency using a complete binary pruning tree. The authors considered fast multi-keyword semantic ranked search in cloud computing [13] as a base work and compared their proposed works with the base work. They proved that the proposed schemes outperform the existing scheme in terms of search efficiency with security.

Joo *et al..*, [14] have proposed a certified Homomorphic encryption process that guarantees the privacy and legitimacy of data simultaneously. Authors have investigated the relationship between different homomorphic security concepts for authenticity and confidentiality in their proposed work. The authors have also developed schemes to support arithmetic circuits which is chosen-ciphertext secure for authenticity and confidentiality. This method is based on approximate error-free GCD assumption.

Xia *et al.*, [15] have implemented a protected, data efficiency and flexible keyword switch based on schema over cloud encrypted data to improve security and limit search time. The project has suggested two modules, the Vector Space Model and the TF x IDF model by creating a distinct tree-based reference structure. Both models and the searchable depth search algorithm help improve keyword search accuracy and reduce search time.

In present era to extract their data from cloud storage, clients select the required services provided by Cloud Service Providers (CSP). Providing required resources and data security to the users are challenging task to CSP. Sonawane *et al.*, [16] have implemented a secure search system without any real data and traps to provide expected output to data users. The newly proposed scheme implemented a user verification process and a secret key protocol to prevent attackers by conducting illegal searches. The authors also used a new method for translating keywords that handle spelling errors. Cloud data users extract their documents from public clouds by adopting cloud paradigm. Providing security to prevent precise data is a overhead for CSP. Although, data search encryption function that make users to search encrypted data securely, which do not support for present data. To overcome these issues Gladiss [17] has proposed an effective system with strict privacy requirements using limited search terms. The newly implemented method enhances the correlation of search outcome. In further work, authors propose to implement communication and computer computing more with advance features.

Fiore *et al.*, [18] have suggested a well-established calculation of encrypted data to enhance the encrypted cloud search efficiency. The Verifiable Computation (VC) protocol uses Fully Homomorphic Encryption (FHE) as a verification tool and also uses the Ad-Hoc protocol to work on very large database. This work adopt tools such as bilinear groups, a modified version of the Brakerski and Vaikuntanatha (BV) homomorphic encryption scheme.

Johnson *et al.*, [19] have suggested a main interpretation of Homomorphic signature safety. In this work, the authors implemented the signature owner of the signature to a small message made illegally. The authors added a pre-signed report and a schema for signing homomorphic agreed for both unions and taking subsets. The authors have proven that the signature system [20] is not safe when the homomorphic process is used to add a total value.

Song *et al.*, [21] suggested the concept of cryptographic systems to overcome search problems in encrypted cloud data and to provide secure evidence for cryptosystems. This application provides controlled search power to a third-party server therefore the cloud server cannot search until it authorizes or authorizes. This schema applies to encrypted queries so that keywords do not expose the server, and support query detection so that the untrusted server does not receive any of that search information about transparency. The proposed schema is simple, quick, and flexible.

Wang *et al.*, [22] have provided an effective follow-up rule to achieve a standardized search that allowed for small leaks of key details in keyword privacy and to explain the problem of secure keyword search [23] in addition to encrypted cloud data. The authors have examined that the standard encryption system works far better than the symmetric encryption schemes that are searchable. They use the encryption process that balances orders to keep keyword commands. And one of the many ways to store planning strategies to support relevant points is to improve computed search.

Catalano *et al..*, [24] have used classics to validate computation in data presented as elements of cryptographic groups. Linearly homomorphic certified encryption and Linearly Homomorphic Signature Schemes are two schemes presented here to ensure security against a weak Random Message Attack (RMA). With this, the authors attain full security guarantees. In the present system, data confidentiality and verification is achieved through a modified homomorphic MAC process with the security and privacy of a multi-word search scheme. The important challenge in these programs is to reduce computer time and improve the performance of the system and at the same time conduct effective and secure searches with encrypted data. Our main concern is to reduce the calculation time spent on index production, file uploading and file search on a cloud server and getting accurate search results.

III. BACKGROUND WORK

A. *Homomorphic MAC*

The Homomorphic MAC encryption process is used to provide external data security and data encryption. This method provides the environment for processing encrypted data in cloud storage. The Homomorphic encryption process can test encrypted data encryption without access to a private key and generate output in an encrypted format. Homomorphic MAC encryption permits users to do computation on pre-verified data with the help of a public key. The tag δ value is used to verify the accuracy of the calculation results. The user's private key is used to verify the actual data and ensure that the δ authorizes the correct computer output.

B. *Hashing Techniques*

Hashing is one of the foremost techniques for converting a set of characters into a predetermined size number and utilized as a key for strengthening security. If we want to restore a document from a cloud data server, we need to search for all index values is an inefficient method. To improve data searching efficiency, we follow the hashing technique [25], which is used to create a reference file containing the hash code of all the documents uploaded to the cloud. This method minimizes the time take to search and improves search efficiency by getting specific results.

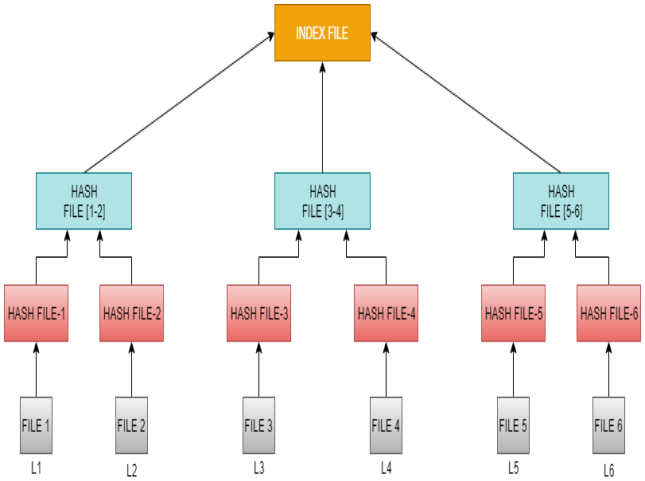

Fig. 1. Hashing Technique in Cloud Computing to generate tag values.

Fig. 1. Demonstrates the hashing process implemented to generate the tag value. A leaf node is defined for each file in the document , and the internal node has a child node label created using the cryptographic hash process. The reference files having the hash value for a specific file that is sent to the cloud based server. There are multiple cryptographic hashing functions, of which Secure Cryptographic Hash function is the most efficient algorithm. This algorithm supports to create a search key map on a real record and improve privacy from a third-party server.

C. Secure Secret Key Generation Technique

The RSA algorithm is implemented to generate a secret key *'SK'*. The algorithm is based on asymmetric cryptography having two keys namely - public key and private key. The data user extracts the files from the cloud server by decrypting the text using a secret key. When data users want to access the document from the cloud server, the he/she sends a request to the data owner. Once the data owner receives the request, he/she verifies the data user by granting a verification key. By utilizing a trapdoor key formed using the verification key and query keyword, data users retrieves the files.

D. Trapdoor Key

Cloud server uses the Trapdoor key to shield search queries from attackers. Input to the trapdoor key generator module [26] is the secret key and the keyword of the query. The output is the trapdoor key from the cloud based server in a highly secured way. Using the trapdoor key, the top-*k* search results are obtained by the data user. Trapdoor activity has different levels, depending on the need. Cryptography widely uses the trapdoor function and for that reason trapdoor key improves data protection by preserving the query keywords from third-party servers in a highly secured form.

IV. PROBLEM STATEMENT AND SYSTEM MODEL

Given a document F with a set of files (F = f1, f2, ... fn) which are to be uploaded by the data owner to the cloud based server. Our objectives are:

- Index computation overhead improvement
- File upload time reduction
- Improve search performance and get accurate search results in a short period of time.

The below fig 2. shows the schematic of the new system model. Three groups are part of the model namely - Data User, Data Owner and S-CSP (*Server side- Cloud Service Provider*).

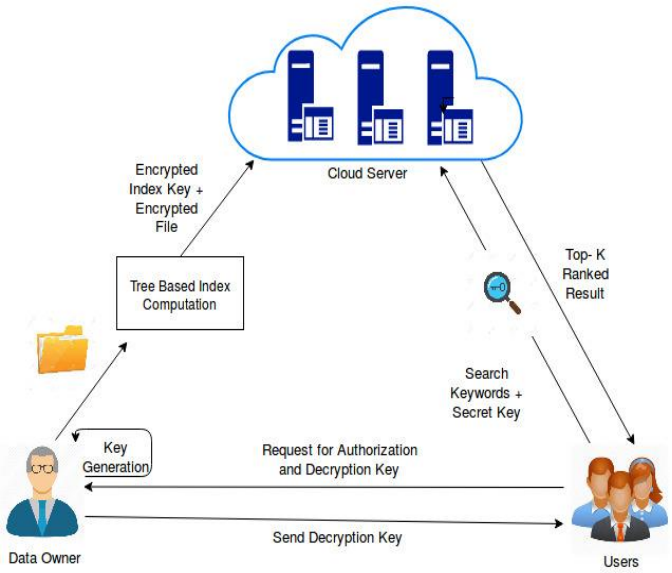

Fig.2. System model

Data users: The user needs to seek permission from the data owner to access authorized data in the cloud by submitting the data user's private key. After receiving the access request from the data user, the data owner verifies the user's identity and decides whether to give data access permission or not to the user. A valid user gets the access permission and public key to download the required document. After receiving the access user gives a public key, and multiple keywords to the cloud server and sets the' k' value to get top 'k' search results. The cloud server provides top search results to the client machine is close to 1. Top 'k' search results helps us get the specific outputs at the top and helps to limit unrelated values. This simplifies the user workload and makes the user work simple and easy.

• *Data Owner*: An individual who stores the original document for the first time in cloud based storage is called data owner. An index tree to the document is generated when the data owner uploads a new set of files. The homomorphic MAC encryption technique is used to encrypt and deploy the index file and the document. By encrypting the documents, the data owner maintains security and a secret key for the outsourced data. Whenever a data user requires to download the files, they must request the key from the data owner. After user authentication, the data owner securely issues the decryption key to the user.

• *Cloud Based Service Provider (CBSP):* The cloud based service provider allows computation capabilities and offers many other services to the data owner and also provides storage space for the data in the cloud. The data owner cannot trust the cloud based server as it is the third-party server. The data owner should provide security and privacy for the stored data as it is the most critical requirement. It is also challenging task for the data owner. To provide data security and confidentiality for the outsourced data, the data is encrypted using the Homomorphic MAC encryption technique before uploading the to the cloud storage by the data owner. Among the operations performed on the cloud based server, searching for a document is the most common operation.

For searching the documents on the cloud data server, the user submits multiple-keywords and the top 'k' value to the cloud server machine as input. After receiving input values from the user, the server will execute the query on the encrypted data and send a result to the user.

V. IMPLEMENTATION

The proposed implemented scheme provides the most relevant search output according to the user requirement by executing a multi-keyword ranked search on encrypted data. To remove redundant search output, the users have set up the top 'k' value manually. We have implemented the relevance score method to support ranked search results. To provide search result efficiency, we have proposed tree-based indexing and Greedy depth-first search techniques and to provide security to the outsourced data the homomorphic MAC encryption technique is implemented. The proposed scheme has *setup files*, *IndexBuild*, *TrapdoorGen*, and *RelevanceScore* stages.

- *Setup:* In the initial stage, the data owner produce the secret key $SK= \{S, M_1, M_2\}$,which contains a randomly created vector value S and invertible matrices M_1 and M_2.
- *IndexBuild:* The index tree is constructed for the document F using *TreeBasedIndex(F).* Then the index tree that is to be uploaded to the cloud based server, is encrypted by the system. $I_u=\{M_1^T D_u', M_2^T D_u''\}$. Where D_u' and D_u'' are random vectors for index vector D_u.
- *TrapdoorGen:* This process performs query search and the input is the secret key. It produces an encrypted query as a trapdoor.
  $TD = \{M_1^1 Q', M_2^1 Q''\}$, Where Q' and Q'' are random query vectors.
- *RelevanceScore:* The cloud server produces the relevant *RScore (Du, Q)* for node u in the index tree, where *Du* is the base vector and Q is the new query vector . The relevance score calculation from encrypted and un-encrypted vector are the same.

A. Tree based index construction

In the proposed system, to create the index tree structure of the uploaded files, a tree based index generation algorithm is used. In this tree structure the first node formed is called a leaf node and all other nodes added to the root node based on its term frequency. The reference tree is designed in the form of clear text. The tree-based index algorithm shown in Table 1. The index is generated using the *GenID*() function, where *Du[i]* always maintains the highest normal frequency value and helps to calculate the most relevance score. A set of newly generated nodes is called *TempNodeList* and a group of current node processed without parent node is called *Current NodeList*.

TABLE 1. TreeBasedIndex(*F*) Algorithm

Input: Document *F=(f1, f2, - - - -, fn)* and File Identifier *FID(1, 2, 3, - - - - , n).*
Output: Index Tree *T.*
1:for each document *FID* in *F* **do**
2: construct a leaf node *x* for *FID, x.ID=GenID(), x.FID = FID, D[i] = TF * FID, x.pl = x.pr =0.*
3: Add *x* to *currentnodeList;*
4: endfor
5: while set of nodes in *currentnodeList* >1 **do**
6: **if** the set of nodes in *currentnodeList= 2h* **then**
7: **for** each node *x', x''* in *currentnodeList* **do**
8: create parent node *x* for *x* 'and *x*''.
9: Add *x* to *tempnodeList;*
10: **endfor**
11: **else**
12: **for** each node x' and *x'* 'of the (*2h-2*) nodes in *currentnodeList* **do**
13: construct a parent node *x* for *x'* and *x*''
14: Insert *x* to *TempNodeList*;
15: **endfor**
16: Create a parent node *x1* for (*2h - 1*) and *2h* node, then initialize a parent node *x* for *x1* and the *(2h + 1)* node;
17: Add *x* to *TempNodeList*;
18: **endif**
19: Replace *TempNodeList= 0* and *CurrentNodeList ->TempNodeList;*
20: endwhile
21: return the node left in *CurrentNodeList*, the root of index tree τ

B. Greedy depth-first search technique

A recurring search method is implemented in the proposed system called the Greedy depth-first search technique [27]. We produce a set of results called as *(RScore, FID)* identified as *RList*. *RList* is the significance list and *RScore* is the significance score of the file f_{FID} to a query. The maximum k number of eligible documents is stored in the *RList*. According to *RScore*, the elements are arranged sequentially down and updated during the search process. The greedy depth first method algorithm is represented in Table 2. Index vector and query vector is used to calculate the *RScore*, the k^{th} rating is set to zero, which is the smallest performance rating in the list. *hchild* and *lchild* are child nodes that are used to store the upper and lower points of a tree node.

TABLE 2. GDFS(IndexTreeNode x)

1:if node $x \neq$ leaf node **then**
2: **if** RScore > *kthscore* **then**
3: **GDFS(***x.hchild***);**
4: **GDFS(***x.lchild***);**
5: **else**
6: **return**
7: **endif**
8: else if RScore > *kthscore* **then**
9: From *RList* delete smallest similar score element.
10: Add a new element and sort *RList*
11: endif
13: return
14: endif

VI. PERFORMANCE EVALUATIONS

In the proposed system, we have used a tree-based indexing system on top of a secure index search system. We found that the efficiency and performance of the proposed system is improved compared to the existing system. A tree-based indexing system has a lesser indexing time and helps to reduce file outsourcing time compared to existing methods.

To build a reference tree structure in the data owner module and a query reference table in the data user module, vector space model and term "frequency (TF) × inverse document frequency (IDF) " model are used. The Vector space Model is also called as the algebraic model which reflect the text document as vectors of identifiers. Each document is represented as a vector while each item in the vector showed the TF value of the keyword in the document. The input query given by user is represented as vector, where the elements in the vector are standard IDF values of the query keyword in the document collection. The cosine angle formed is used to calculate the points of similarity between the documents and the query. The term frequency and the number of conflicting documents are used to determine how often a word is repeated in a single collection text. To determine the term frequency, we check each document and calculate how often each word appears. To find the Inverse Document frequency we divide the size of the text collection by the number of documents containing the keyword. Finally, we constructed a searchable index structure with a vector space model and obtained a cosine and TF x IDF measurement to obtain statistical results.

The performance of proposed system was demonstrated through a test set executed on a laptop which was equipped with an Intel-Icore i5 CPU [2-core 2.2GHZ], 2TB RAM using 64bit Win 10 OS and Matlab R2016a.

A. Index Construction

TABLE 3. Index generation time of VPS and EFURMS.

SL No.	Number of Documents Size in mb	VPS Time in ms	EFURMS Time in ms
1	1	20	10
2	2	55	28
3	3	60	42
4	4	103	80
5	5	250	140
6	6	360	208
7	7	450	298
8	8	560	328
9	9	620	425
10	10	750	480

Table 3. Shows the time consumed to construct a index tree file for different numbers of documents with various size by the existing and the proposed system. Here we consider document size in mb and index generation time in milliseconds. documents with small size consumes less time in both VPS and EFURMS.

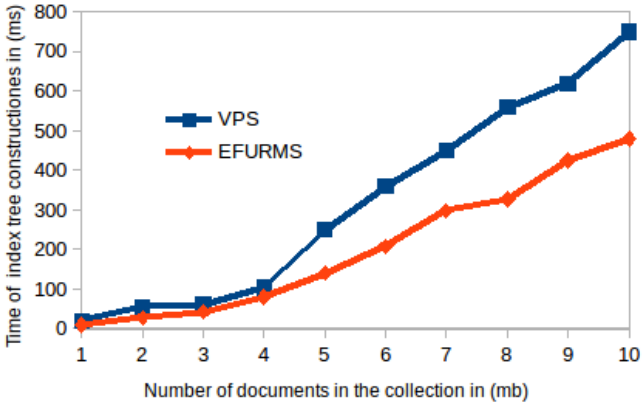

Fig. 3. Comparison of Time taken for index file computation in existing and proposed system.

Fig. 3 shows the graph of time taken to construct a index tree for different numbers of documents. The suggested scheme reduces the index construction time as they use a tree-based index algorithm. The construction of the index value and the construction of the index tree structure are one of the most critical steps. In the current system, generating an index file for the files uploaded by the data owner to the cloud based server is more time intensive. To reduce this time, a tree-based indexing algorithm is used in the new system. The proposed algorithm takes n-number of files and divides the content of the file into '*K*' number of keywords i.e. Vector names from the files, which will encrypt and upload the index file to provide data security.

B. Upload Time

TABLE 4. Document upload time of VPS and EFURMS.

SL No.	Number of Documents Uploading Size in mb	VPS Time in (x 100 ms)	EFURMS Time in (x 100 ms)
1	1	140.72	129.72
2	2	150.01	149.93
3	3	252.5	221
4	4	300.1	280.02
5	5	381.2	360.21
6	6	321.45	292.01
7	7	329.36	302.54
8	8	360.48	348.01
9	9	421.68	410.68
10	10	599.36	528.36

Table 4. Shows the time consumed to upload set of encrypted index file and documents with various size to the cloud storage by the data owner. Here data owner uploading encrypted documents using Homomorphic MAC encryption process which reduces the uploading time The EFURMS scheme aids in reducing the file size and makes uploads faster. Therefore EFURMS scheme provides better results compared to VPS scheme.

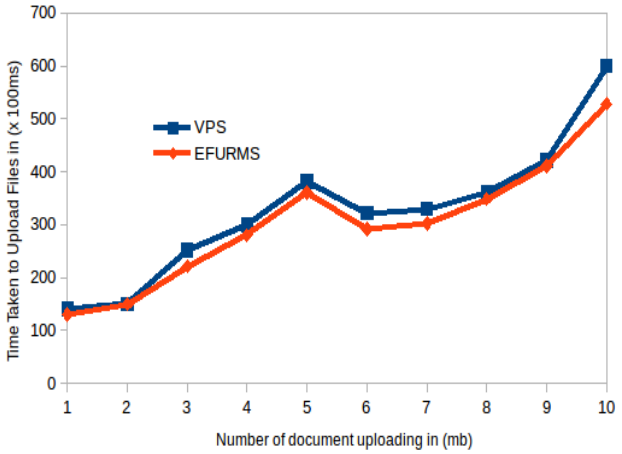

Fig. 4. Comparison of the time for uploading of documents to in the proposed and existing system.

As seen in Fig. 4, there is a slight reduction in the time taken to upload files to the cloud based server. The new scheme aids in reducing the file size and makes uploads faster. The proposed scheme therefore provides superior performance results compared to existing scheme.

If the owner of the data uploads the file to the cloud based server directly, it takes more time even if the Internet connection is fast. If the content of the index file is large, it increases the time to upload the document. Therefore, the index file is created only for the keywords that are found in each file. Index file and all files will be uploaded to a encrypted cloud server using the Homomorphic MAC encryption process. This reduces the file size as the content in each file is encrypted from thread to number format. So it reduces the time to upload files to a cloud based server compared to files with normal encryption.

C. Search Time

TABLE 5. Document Retrieval time of VPS and EFURMS.

SL No.	Number of Retrieved Documents	VPS Search Time in ms	EFURMS Search Time in ms
1	1	6	2
2	2	9	5
3	3	18	10
4	4	25	15
5	5	36	29
6	6	48	32
7	7	46	39
8	8	69	56
9	9	83	79
10	10	106	91

Table 5. Shows the time consumed to retrieve documents with various size from the cloud storage by the data owner/ user. By implementing the advance techniques the proposed scheme can achieve better search time compare to the existing scheme.

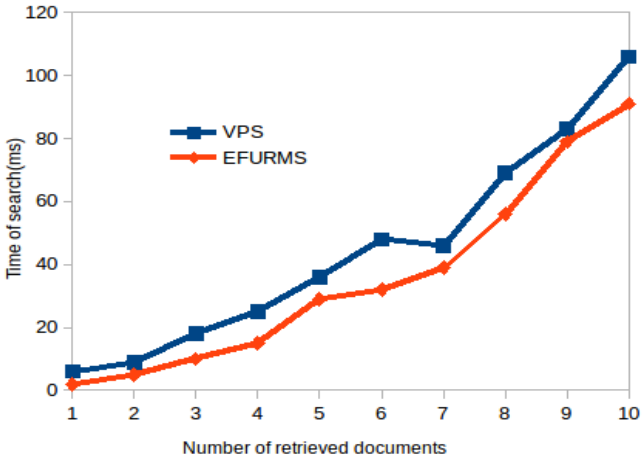

Fig. 5. Efficient Search functionality with five key values.

One of the most widely used functions in the cloud based server is search. Either a single keyword or multiple keywords can be used by the data user to search. To increase the search speed, the Greedy depth-first search technique is applied with a tree-based index calculation method. This provides more accurate search results in less time. Searching a document using multiple keywords yield quicker search results as shown in Fig. 5.

VII. CONCLUSIONS

In this paper, a EFURMS scheme is implemented which generates and stores the index values to the files that need to be uploaded to the cloud based server, using a tree based index generation. This technique is found to reduce the index computation time and the time taken to upload the document. Implementation of the Greedy depth-first search algorithm and analysis of the results highlight that it is the most effective way to get accurate search results. The proposed scheme designed in such a way that user can give multi keywords as a input to the search engine. It make searching process more faster and can able to produce most accurate and top-*'k'* search results to the user. The proposed scheme features helps to protect data and provide expected output to the user efficiently with less time. However, these methods work best for large datasets such as big data. Still users are facing many issues in multi-keyword search process. Software and Service Engineering (SSE) schemes often assume that all users are trustworthy. An un-trusted user will lead to many security issues, they may hack encrypted documents and share it to unauthorized users. Additionally, an un-trusted user can send his or her secure keys to unauthorized users. In future work, we will try to improve the SSE system to manage these challenges by implementing advanced techniques .

REFERENCES

1. R. Buyya, R. N. Calheiros, J. Son, A. V. Dastjerdi, and Y. Yoon, "Software-Defined Cloud Computing: Architectural Elements and Open Callenges," *International conference on advances in computing, communications and informatics (ICACCI)*, pp. 1–12, IEEE, 2014.
2. Z. Ying, H. Li, J. Ma, J. Zhang, and J. Cui, "Adaptively secure ciphertext-policy attribute-based encryption with dynamic policy updating," Sci China Inf Sci, vol. 59, no. 4, pp. 042701:1–16, 2016.

3. N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data," *IEEE Transactions on parallel and distributed systems*, vol. 25, no. 1, pp. 222– 233, 2013.
4. Chen, Li, Xingming Sun, Zhihua Xia, and Qi Liu. "An efficient and privacy-preserving semantic multi-keyword ranked search over encrypted cloud data." *International Journal of Security and Its Applications*, vol. 8, no. 2, pp. 323-332, 2014.
5. X. Li, D. Chen, C. Li, and L. Wang, "Secure Data Aggregation with Fully Homomorphic Encryption in Large-Scale Wireless Sensor Networks," *Sensors*, vol. 15, no. 7, pp. 15952–15973, 2015.
6. Peralta, Goiuri, Raul G. Cid-Fuentes, Josu Bilbao, and Pedro M. Crespo. "Homomorphic Encryption and Network Coding in IoT Architectures: Advantages and Future Challenges." *Multidisciplinary Digital Publishing Institute*, vol. 8, no. 8, pp. 827, 2019.
7. Goyal, Prachi. "Optimizing System Efficiency and Performance using Tree Based Indexing Scheme." In *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, pp. 537-541, 2019.
8. Ding, Xiaofeng, Peng Liu, and Hai Jin. "Privacy-Preserving Multi-Keyword Top- k Similarity Search Over Encrypted Data." *IEEE Transactions on Dependable and Secure Computing* , vol.16, no. 2, pp.344-357, 2017.
9. Jiang, Shan, Jiannong Cao, Julie A. McCann, Yanni Yang, Yang Liu, Xiaoqing Wang, and Yuming Deng. "Privacy-Preserving and Efficient Multi-Keyword Search over Encrypted Data on Blockchain." *IEEE International Conference on Blockchain (Blockchain)*, pp. 405-410, 2019.
10. Jivane, Anjali Baburao. "Time efficient privacy-preserving multi-keyword ranked search over encrypted cloud data." *IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, pp. 497-503, 2017.
11. Peng, Tianyue, Yaping Lin, Xin Yao, and Wei Zhang. "An efficient ranked multi-keyword search for multiple data owners over encrypted cloud data." *IEEE Access, vol.* 6, pp. 21924-21933, 2018.
12. Dai, Hua, Yan Ji, Geng Yang, Haiping Huang, and Xun Yi. "A privacy-preserving multi-keyword ranked search over encrypted data in hybrid clouds." *IEEE Access* , vol.8, pp. 4895-4907, 2019.
13. Y.Yang,J.Liu,S.Cai,andS.Yang,''Fastmulti-keywordsemanticranked search in cloud computing,'' Chin. J. Comput., vol. 40, pp. 158–171, Jun. 2017.
14. C. Joo and A. Yun, "Homomorphic Authenticated Encryption Secure Against Chosen-Ciphertext Attack," *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 173–192, Springer, 2014.
15. Z. Xia, X. Wang, X. Sun, and Q. Wang, "A Secure and Dynamic MultiKeyword Ranked Search Scheme over Encrypted Cloud Data," *IEEE transactions on parallel and distributed systems*, vol. 27, no. 2, pp. 340– 352, 2015.
16. Z. Guo, H. Zhang, C. Sun, Q. Wen, and W. Li, "Secure Multi-Keyword Ranked Search over Encrypted Cloud Data for Multiple Data Owners," *Journal of Systems and Software*, vol. 137, no.1, pp. 380–395, 2018.
17. G. adn Merlin. N, "Fuzzy Based Implementation of Multi - Keyword Ranked Search over Encrypted Cloud Data in Secure Cloud Environment," *in International Journal of Scientific Research and Management (IJSRM)*, pp. 7610–7617, IJSRM, 2017.
18. D. Fiore, R. Gennaro, and V. Pastro, "Efficiently Verifiable Computation on Encrypted Data," *in Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 844–855, ACM, 2014.
19. R. Johnson, D. Molnar, D. Song, and D. Wagner, "Homomorphic Signature Schemes," *in Cryptographers Track at the RSA Conference*, pp. 244–262, Springer, 2002.
20. Chang, Jinyong, Hui Ma, Anling Zhang, Maozhi Xu, and Rui Xue. "RKA security of identity-based homomorphic signature scheme." *IEEE Access*, vol. 7, pp. 50858-50868, 2019.
21. D. X. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," *in Proceeding 2000 IEEE Symposium on Security and Privacy.*, pp. 44–55, IEEE, 2000.
22. C. Wang, N. Cao, K. Ren, and W. Lou, "Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data," *IEEE Transactions on parallel and distributed systems*, vol. 23, no. 8, pp. 1467–1479, 2011.
23. Z. Fu, X. Sun, Q. Liu, L. Zhou, and J. Shu, "Achieving Efficient Cloud Search Services: Multi-Keyword Ranked Search over Encrypted Cloud Data Supporting Parallel Computing," *IEICE Transactions on Communications*, vol. 98, no. 1, pp. 190–200, 2015.
24. D. Catalano, A. Marcedone, and O. Puglisi, "Authenticating Computation on Groups: New Homomorphic Primitives and Applications," *in International Conference on the Theory and Application of Cryptology and Information Security*, vol. 8874. pp. 193–212, Springer, 2014.
25. Sujito, Sujito, and Wildan Mahir Muttaqin. "Hashing Variable Length Application For Message Security Communication." *ARPN Journal of Engineering and Applied Sciences*, vol. 14, no. 01, pp. 259-264, 2019.
26. Döttling, Nico, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. "Trapdoor hash functions and their applications." In *Annual International Cryptology Conference*, vol. 11694, pp. 3-32. Springer, 2019.
27. Xia, Zhihua, Xinhui Wang, Xingming Sun, and Qian Wang. "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data." *IEEE transactions on parallel and distributed systems*, vol. 27, no. 2, pp. 340-352, 2016.

AUTHORS PROFILE

Mrs. Bhavya M received the Bachelor of Engineering degree in Computer Science and Engineering from The University Visvesvaraya College of Engineering Bangalore, in 2010, and the Master of Technology in Computer Science and Engineering from Cambridge Institute of Technology Bangalore, in 2014, Visvesvaraya Technological University, Belgaum, India. She is currently working toward the PhD degree from Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore, Bangalore University, India. Her research interests include cloud computing, Cloud Data storage management and data security. She is a member of IEEE since 2015. She has published research papers in reputed international journals and conference. She has one years of teaching experience, One year of Industrial experience and 5 years of Research Experience.

Dr. Thriveni J has completed Bachelor of Engineering, Masters of Engineering and Doctoral Degree in Computer Science and Engineering. She has 4 years of industrial experience and 25 years of teaching experience. Currently she is Professor in the Dept. of CSE, University Visvesvaraya College of Engineering, Bangalore. She has more than 100 research papers to her credit. She has produced six doctorate students and guiding 07 PhD Students. Her research interests include Networks, Data Mining, Cloud Computing and Internet of Things.

Dr. K. R. Venugopal is currently the Vice Chancellor Bangalore University, Bengaluru. He obtained his Bachelor of Engineering from University Visvesvaraya College of Engineering. He received his Masters degree in Computer Science and Automation from Indian Institute of Science Bengaluru. He was awarded Ph.D. in Economics from Bangalore University and Ph.D. in Computer Science from Indian Institute of Technology, Madras. He has a distinguished academic career and has degrees in Electronics, Economics, Law, Business Finance, Public Relations, Communications, Industrial Relations, Computer Science and Journalism. He has authored and edited 64 books on Computer Science and Economics, which include Petrodollar and the World Economy, C Aptitude, Mastering C, Micro-processor Programming, Mastering C?? and Digital Circuits and Systems etc., He has filed 101 patents. During his three decades of service at UVCE he has over 640 research papers to his credit. His research interests include Computer Networks, Wireless Sensor Networks, Parallel and Distributed Systems, Digital Signal Processing and Data Mining. He is a Fellow of IEEE, ACM and ISTE.