International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1, October, 2019

Verifying Fuzzy Keyword Search Over Cipher Text in Cloud Computing

P. Adlene Ebenezer, Sweta Pasayat, Sunny Sunan

***Abstract**: To enhance the potency of knowledge looking out, most knowledge house owners store their knowledge files in numerous cloud servers within the kind of ciphertext. Thus, economical search victimization fuzzy keywords become a vital issue in such a cloud computing atmosphere. Searchable cryptography will support knowledge user to select and retrieve the cipher documents over encrypted cloud knowledge by keyword-based search. Most of the prevailing searchable encryption schemes solely specialize in the precise keyword search. When knowledge user makes writing system errors, these schemes fail to come to the results of interest. In searchable encryption, the cloud server may come to the invalid result to knowledge user for saving the computation price or alternative reasons. Therefore, these precise keyword search schemes notice very little sensible significance in real-world applications. So as to deal with these problems, we tend to propose unique verifiable fuzzy keyword search theme over encrypted cloud knowledge. We tend to propose a verifiable precise keyword search theme which extend this theme to the fuzzy keyword search theme. Here we tend to thus propose a system for fuzzy keyword sets rather than precise word search. This will help us drastically to reduce the costs and it also allows to have multi-users using the system simultaneously.*

Keywords: Cipher-text, Fuzzy Keywords, Encryption, Linked List, Jaccard Similarity

I. INTRODUCTION

Cloud Computing is one of the most rapidly growing technologies in the current scenario. More and more data are stored every day in the cloud. Basically, the term cloud refers to the process of employing a network of remote servers hosted on the web to store, manage and method knowledge, instead of a neighborhood server or a private personal computer. Basically, what happens when we store the data in a cloud-based service is that the whole of the data is stored in the web servers and then we can retrieve the data as and when required. So basically, what we intend to do is that many at times it happens that while searching on the net, we tend to either misspelt the word or we write the word incorrectly, most of the times it happens that the results we get are not what we searched for This is the problem statement of our paper which we intend to solve.

Revised Manuscript Received on October 30, 2019.

* Correspondence Author

Ms. P. Adlene Ebenezer*, Assistant Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India.

Sweta Pasayat, UG Scholar, B. Tech, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India.

Sunny Sunan, UG Scholar, B. Tech, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India.

We desire to overcome the problem of the existing schemes suffer i.e. suffering from efficient handling of multi-letter errors or cannot distinguish anagrams. There have been numerous attempts to solve this problem.

We have chosen the one of the most recent papers on this topic in this paper they have used the Jaccard Similarity and the monogram set to get to the desired results. Basically, Jaccard Similarity ensures maximum similarity for nearest word possibly the original one and the term monogram refers to the original word's position in the word which enables to find out original word from its typo with maximum similarity possible. So, what we are trying to do here is to refine the already done work. We intend to this with the help of the employment of the linked list as our source index to achieve the efficient storage. So, what we have done is that we have constructed a linked list as our source index to achieve the efficient storage. We take the keyword from user and then form the linked list with 3 nodes in each, and then we generate and store the keywords and after that all the processes like keyword extraction and the algorithm is used and then finally it is sent to the user as the result. We can see this with an example as using Jaccard Similarity Algorithm we do the keyword extraction process. For example, we are setting the keyword for Basketball Ground as 'BG' and at the same time we also set the keyword for Background as 'BG'. What Jaccard similarity in our case of problem statement does is it checks the similarity in different sets for the 'BG' keyword. Therefore, it derives from Jaccard similarity which words are most frequently used words related to the keyword and moreover derives those files from the cloud storage. So, this is how we have implemented this particular algorithm here. So, in this way the whole system works.

II.LITERATURE SURVEY

The concept of keyword search over encrypted data in cloud has evolved tremendously over the past decade. Several techniques and procedures are introduced like in [1] which uses synonym-based search which supports only a group of users. Moreover, this technique allows only authorized cloud consumers. The VSM technique used in [1] measure the similarity between the query vector and file index vector and thus produces accurate results. The enhancements that can be made in [1] is that the use of semantics-based research which supports syntactic transformation and Natural Language Processing technology. In [2], a fast and secure technique called trap - door construction is used but the whole process becomes tedious for cloud with large users. Moreover. The above-mentioned process focuses largely on the cipher text and less on the whole process.

The edit distance technique which uses the Guess algorithm in [2] is said to reduce the security of the search but the clever adversary constructed breaks its provable security reduction. A F2SE search scheme was designed in [3] which uses keyword fingerprint extraction algorithm and secures with KNN encryption method. The memory cost of F2SE is comparatively lower than other wildcard schemes like SFKS scheme. The relationship between the time taken for searching on the indices and the number of keywords and its magnitude is linear. As it has lower memory storage capacity and other demerits, it was optimized to support matching rules and functionalities of fuzzy multi-keyword search later. Data discovery and user searching experience was attempted to improve in [4] by imposing a secure ranked search where the overhead of generating the trapdoor is eliminated by directly querying the cloud server. It uses a keyword dictionary which is formed and it is expanded without altering the contents in the original dictionary. This method proves to be more efficient as in RFMS time taken to generate indices at data owner is decreased as the data user directly queries the cloud server. In RFMS, overhead of key exchange between user of data is found. In [5], the issue of fuzzy keyword set construction was tackled which improved the system usability and user's retrieval efficiency. The fuzzy keyword set procedure made more precise keyword searches and it also used the feedback scheme which produced better results even if keywords are misspelled because of the flexible pointer used in retrieval process. The pointer vector used directs the set which has higher rank and therefore the files having more relevant keywords are displayed first

III. ARCHITECTURE DIAGRAM

The architecture diagram is shown below. It has been divided into 4 modules. Starting from the process of key generation which then goes to the linked list process where it is basically divided into a linked list with that of 3 nodes which is one of the most important features of our work process. Next, we take the keyword from the user end and then we process it to the document storage unit. After that we use our Jaccard Similarity algorithm where the keyword extraction takes place and then it is converted into the text file having the .txt format. Then finally the documents are encrypted and the key is stored into the cloud server where it is then presented to the user. This proposed architecture diagram basically consists of 4 modules with three major parts i.e. cloud server, user end (the person who is searching) and the organization or the person that owns the cloud. The above architecture diagram gives an overview of how actually our proposed system for the given problem statement works. So basically, the four modules that are present can be divided as: 1- Keyword generation 2- Encryption of documents and the keys present 3- Keyword extraction 4- Retrieval of the desired file from the cloud server. Here we use the data structure i.e. linked list with 3 nodes each and the algorithm being used is Jaccard Similarity.

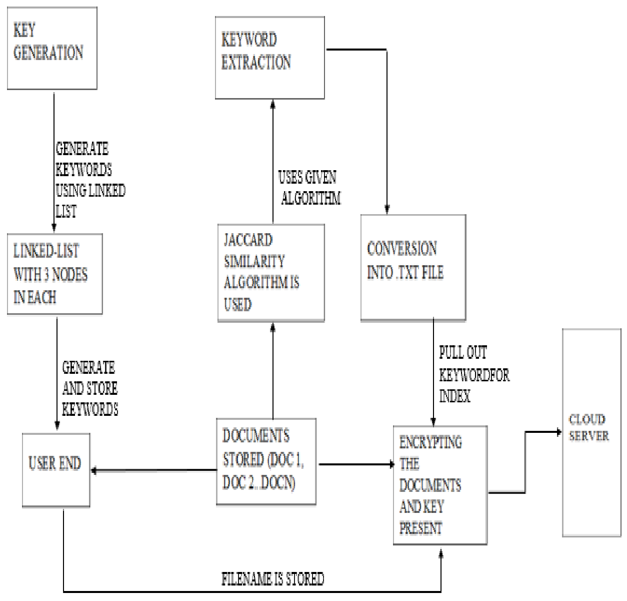
KEY GENERATION
GENERATE KEYWORDS USING LINKED LIST
LINKED-LIST WITH 3 NODES IN EACH
GENERATE AND STORE KEYWORDS
USER END
KEYWORD EXTRACTION
USES GIVEN ALGORITHM
JACCARD SIMILARITY ALGORITHM IS USED
DOCUMENTS STORED (DOC 1, DOC 2...DOCN)
CONVERSION INTO .TXT FILE
PULL OUT KEYWORDFOR INDEX
ENCRYPTING THE DOCUMENTS AND KEY PRESENT
CLOUD SERVER
FILENAME IS STORED

fig 1.1

IV. KEYWORD GENERATION

If we look into how does this module actually work? We can get at brief idea by the given flow diagram provided. This briefs us about the entire process of keyword generation. The workflow in the given flow diagram shows how exactly the keyword is generated from the file to be extracted from the cloud server. This is a major module as keywords are the most necessary thing for any file extraction from file. Here the term linked list plays one of the most important roles, and the term linked list actually refers to an arranged combination such that all the elements have some sort of link with either its predecessor or successor. In most of the cases, it has connection with the successor only. It is further divided into the three major types which are given as Singly, Circular and Doubly. Each one has its own importance and these are further divided into number of types. The working of the linked list is with the help of nodes which contain a data link and the link which references to either to the next node or the previous ones in the list When we see the first module i.e. keyword generation, we see that we take the words from which the files have to be derived. Using the linked list structure with three nodes each the keywords are generated by taking into account the first alphabets of the particular word or set of words. These keywords being generated are stored in the linked list using the index of linked list, this process therefore makes the retrieval and searching of a particular keyword very easy. The keys generated are stored at the user end. The retrieval is made easy since the whole of the linked list is tagged with an index number so that when and wherever we need that word, we just need to go for the index number and the word will be there. Here it is stored in 3 nodes which means the whole of the keyword is divided into 3 components with each having its own set of words to contain. So, this is how the process of keyword generation works. Next, the process of encryption is seen.

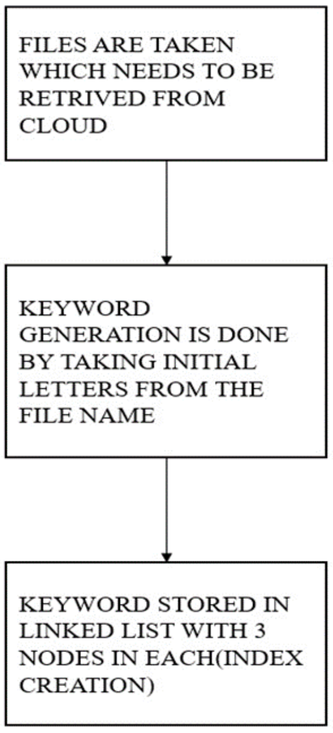
FILES ARE TAKEN WHICH NEEDS TO BE RETRIVED FROM CLOUD
KEYWORD GENERATION IS DONE BY TAKING INITIAL LETTERS FROM THE FILE NAME
KEYWORD STORED IN LINKED LIST WITH 3 NODES IN EACH(INDEX CREATION)

fig 1.2

V. ENCRYPTION OF KEYS AND FILES PRESENT

The term Encryption refers to the process of changing data or information into a code, particularly to forestall unauthorized access. In the Layman terms, it can be defined as a process in which the data is stored in such a way that its contents are not visible to everyone and it's limited to the one which has the key to the access of this data. Encryption of data is very useful for having a secure environment for the files that you deal with in your day today life. This helps to make your files readable and cipherable. Moreover, one can choose with whom and up to what level one wants to share the information. Encryption of data in the cloud makes sure that your sensitive information is secured within layers which is only accessible by you and no one else. This entire concept of encryption has a counterpart called decryption in which we can easily convert our encrypted file to normal format. The flow chart given shows the process in which we encrypt and decrypt our files in cloud. The word cipher means the process of encoding it into an encrypted manner.

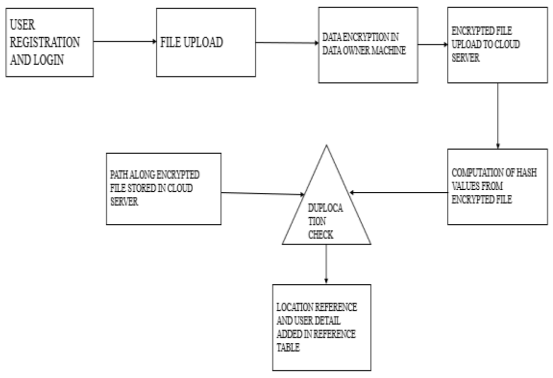
USER REGISTRATION AND LOGIN
FILE UPLOAD
DATA ENCRYPTION IN DATA OWNER MACHINE
ENCRYPTED FILE UPLOAD TO CLOUD SERVER
COMPUTATION OF HASH VALUES FROM ENCRYPTED FILE
DUPLOCA TION CHECK
PATH ALONG ENCRYPTED FILE STORED IN CLOUD SERVER
LOCATION REFERENCE AND USER DETAIL ADDED IN REFERENCE TABLE

fig 1.3

Next comes the second module in which the documents and the keywords that we have generated are stored in the cloud server that is owned by a particular organization or a specific person. For this we can use the zero-knowledge cloud storage such as Sync.com. In such cloud storage we can encrypt our files by just signing in and moreover they cannot be decrypted by anyone else because of the fact that there is a unique password set for decryption which is also attached with you signing in into that particular cloud storage. After that the particular keyword u search for the similar files will show up and moreover you can decrypt the file using the password set by you. Other slightly less easy zero knowledge encryption services are available such as Box Cryptor. These provide one click solution to the problem of encryption and these are absolutely free of cost which help us to reduce the cost. Next, we move to the Keyword Extraction.

VI. EYWORD EXTRACTION

The process of keyword extraction is one of the most important process of this paper. As the name suggests this refers to the process of extracting the keyword for further processing. Whenever we enter a keyword there are many similar kinds of keywords present. Therefore, our aim must be to retrieve such files which are most commonly accessed. The files which come as output should occur according to the frequency of search. The most common example to understand this logic is to imagine how google search works. Basically, when we type a letter what shows us is the most frequent searches. Moreover, when we go on typing the word more prominently our search refines more and more. Therefore, in the following module we use a specific algorithm which is described below and the mathematical formula for the following is even introduced.

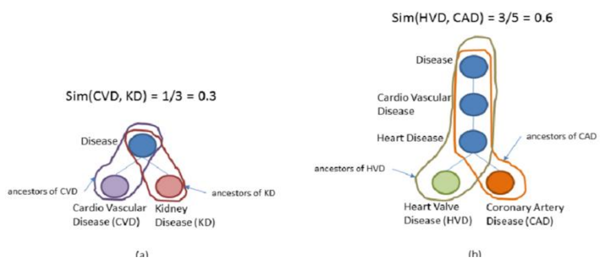

fig 1.4

Then comes the turn of the third module in which we have keyword extraction. Here for the keyword extraction we use an algorithm known as Jaccard similarity. Jaccard Similarity is also known as the Jaccard Coefficient Similarity. What this algorithm suggests is that it compares the data of two particular sets to find out what percentage of data are unique and what percentage of data is similar. Basically, this algorithm finds the similarity percentage between different sets and ranges is from 0% to 100%. Now the question is how does this algorithm actually work in our problem statement. using Jaccard similarity algorithm we match the keyword set with other sets and find out which keywords are maximum in use and shows the not so distinct keyword as result. Hence this is how keyword extraction is done.

VII. JACCARD SIMILARITY

Jaccard Similarity is also known as Jaccard Coefficient. This shows us the similarity between 2 or more given sets. These sets can be of anything. In this case we are using sets of keywords

When we take 2 sets namely "X" and "Y". Let these two given sets be finite sets. Therefore, Jaccard Coefficient is given by J (X, Y) for the aforementioned sets

$$J(X, Y) = |X \cap Y| / |X \cup Y|$$

{equation (1)}

If we want to convert Jaccard Similarity to Jaccard Distance then we can implement the following given formulae

Jaccard Distance $(X, Y) = 1 - J(X, Y)$

{equation (2)}

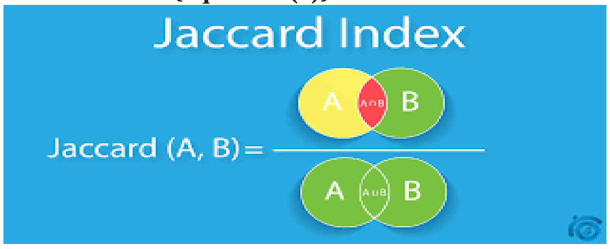

fig 1.5

VIII. RETRIVAL OF DESIRED FILES

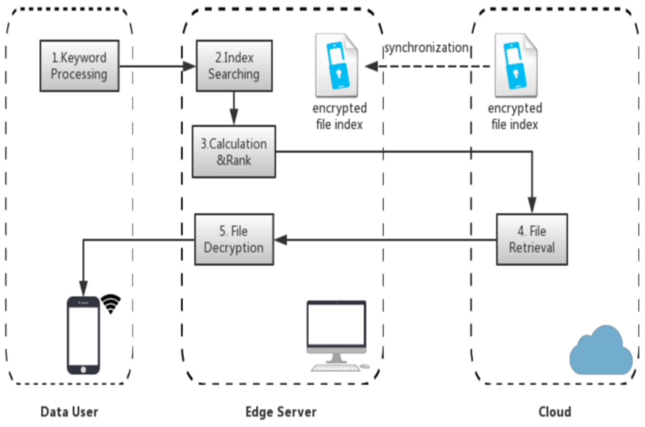

fig 1.6

The last and final module that we come across in our particular architecture diagram is that of retrieval of the desired file from the cloud server. The term cloud server refers to the storage which is available online and can be used to store the data online. This means that the data is not stored locally but it is stored in the internet. It has number of advantages but one of the most important is that the data can be accessed from when and wherever required. Here the whole process is divided into three parts starting from the Data User, Edge Server and the last one is cloud. Here the last part will be in focus. In the above step already the keyword extraction is done. Therefore, from those particular set of keywords the documents that have been stored in the cloud are matched and hence all the similar kinds of files are shown. Such a process helps us in retrieval of files even though they do not match the exact keywords. The Encryption of the file Index is one of the most important process since then only it helps to get to the desired result.

VIII. ALGORITHM

Input:	a: fingerprint index tree G_W
	b: arrange the tree G based on indexing value
Output:	a sorted top-k ciphertext document $C' = (c_1, c_2, \dots, c_k)$
Step 1:	Initialize the set U of capacity k to be null
Step 2:	WHILE (the number of documents in U is less than k)
	Calculate the Jaccard distance Jwi between
	Document is inserted in U in an increasing order of the values of Jwi
	END WHILE
Step 3:	Assign the maximum value of Jwi in U to J_{max}
Step 4:	FUNCTION Search Tree(a)
	Access node a
	B = the first child node of node a
	WHILE (b exists)
	Calculate the Jaccard distance Jwi
	Let $\hat{J}$ = [Jwi,{FIDij,OPE(ek,Scoreij)}]
	IF b has not been accessed & $\hat{J} \leq J_{max}$
	IF b is a leaf node & Jwi < J_{max}
	Node b is inserted in U in an increasing order of the values of Jwi
	Assign the maximum value of the Jwi in U to J_{max}
	ELSE Search Tree(b)
	END IF
	END IF
	B = the next child node of node a
	END WHILE
	END FUNCTION
Step 5:	FOR each FIDij ∊ U
	Sort Jwi as well as OPE(ek,Scoreij) from small to large
	Get $FID' = (FID_1, FID_2, \dots, FID_k)$
	END FOR
Step 6:	Find cipher document set $C' = (c_1, c_2, \dots, c_k)$ that correspond to $FID' = (FID_1, FID_2, \dots, FID_k)$
Step 7:	RETURN $C' = (c_1, c_2, \dots, c_k)$
Step 8:	DECRYPTION $C' = (c_1, c_2, \dots, c_k)$

IX. PERFORMANCE EVALUATION

Performance evaluation refers to the process of analyzing the solution and comparing it with to the previous results so that one can find out how much the results have improved. It will be done on the basis of two measures which are Precision and Recall, these both are one of the most used evaluation criteria to measure the extent of performance. Both of the formulas are listed below-

Precision= number of related files /number of files returned

Recall = number of related files / numbers of relevant documents

Using the above of the two evaluation systems, we can go for the performance evaluation. At first, in the precision equation, it is seen the number of related files which are retrieved by the system and then it divides it with the total number of files returned by the system.

So, to increase the performance evaluation, the number of related files retrieved must be as higher as possible, so that the ratio as overall increases. Here the term related files refer to the matching terms which we need to get. For example- If one searches for the word ‘FOOTBALL’, then by using this precision formula, it should return as much words as close to the word FOOTBALL possible. So, as much as the number of related words are increased, the performance goes on increasing. So, in this way the performance evaluation is tested. Next, the second equation which is the recall evaluation criteria is seen, this refers to the formula which is given by the division which occurs between the number of related files retrieved by system to the total number of relevant documents.

This is almost same as that of the first equation and the only difference being in that it works on the base of the total number of relevant documents instead of the total number of files returned by the system. Next, the results for both of them are seen below-

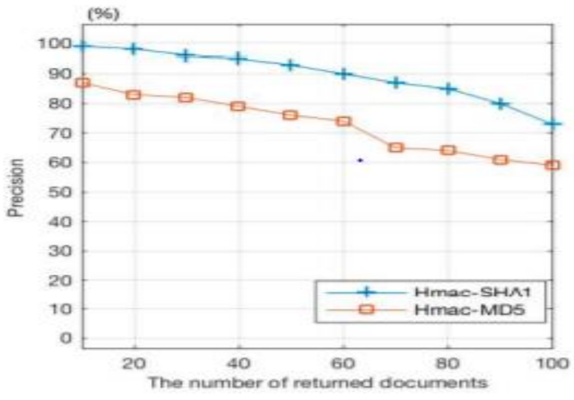

fig: 1.7

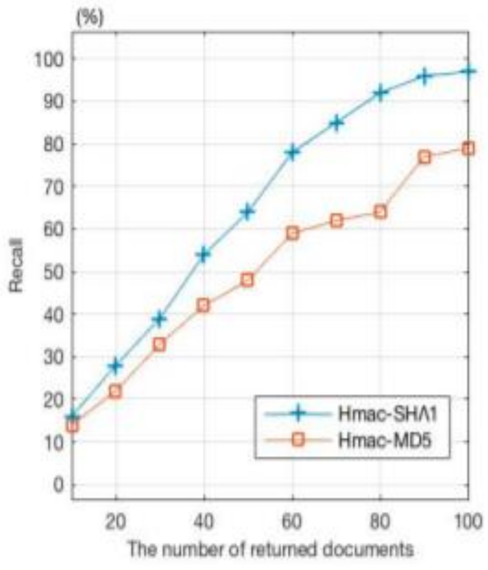

fig: 1.8

So, by these graphs it can be seen that the results have improved in both the cases. So, this is the performance evaluation of our proposed system.

X. CONCLUSION

The conclusion basically defines the whole process of this paper getting published. It first contains a short description of the whole process what is going inside the paper. It started with the keyword of Cipher text, this refers to the encrypted data which means the data which is secured by one method or another and it is protected which can be opened by only those which have the key to open the system. Then it has the literature survey which contains all the references from where the references have been taken. Then it consists of the Architecture Diagram which has a number of components which forms the base of this paper, it has got all the modules which are needed for the working of the system, this basically consists of the a number of modules like keyword generation, then the encryption of keys and files present, this then leads us to the keyword extraction and then we have got the algorithm which we have used, along with that of the Jaccard Similarity. So, after this the performance evaluation has been done by using the two formulas [6], which then leads us to the two resultant graphs which show that there is particular improvement in the results. So, it can be concluded that by using our method there is an improvement as compared to the previous tried methods, this is possible
because we have used the Jaccard Similarity along with the three nodes linked list which leads us to the better execution, better retrieval from storage and finally better pricing. So, in this way the contents of the paper are shown and the results are also generated and shown.

REFERENCES

1. Zhangjie Fu, Xingming Sun, Zhihua Xia, Lu Zhou and Jiangang Shu, “Multi -keyword Ranked Search Supporting Synonym Query over Encrypted Data in Cloud Computing “in IEEE 2013
2. Minghui Zheng and Huihua Zhou,” An Efficient Attack on a Fuzzy Keyword Search over Encrypted Data” in IEEE ICEUC’13,2013
3. Dongsheng Wang, Shaojing Fu and Ming Xu, “A Privacy-preserving Fuzzy Keyword Search Scheme over Encrypted Cloud Data”, in IEEE ICCCTS’13,2013
4. Neelam S. Khan, Dr. C. Rama Krishna and Anu Khurana, “Secure Ranked Fuzzy Multi-Keyword Search over Outsourced Encrypted Cloud Data” in IEEE ICCCT’14,2014
5. Wang Jie , Yu Xiao , Zhao Ming , Wang Yong ,”A Novel Dynamic Ranked Fuzzy Keyword Search Over Encrypted Data” in IEEE 2014 ICDASC
6. Jingsha He, Jianan Wu,Nafei Zhu and Muhammad Salman Pathan , “MinHash-Based Fuzzy Keyword Search of Encrypted Data across Multiple Cloud Servers”

AUTHORS PROFILE

Ms. P. Adlene Ebenezer, Assistant Professor, Dept of CSE, SRM Institute of Science and Technology, Chennai, India Holds a post-graduation degree M.E (CSE) from Anna University Having more than three years of teaching and Industrial. Also, a member of the CSI Chapter experience. She can be reached at Email: adleneep@srmist.edu.in

Sweta Pasayat, UG Scholar, B Tech, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India. She can be reached at Email: sweta_pasayat17@srmuniv.edu.in

Sunny Sunan, UG Scholar, B Tech, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India. She can be reached at Email: sunny_sunan17@srmuniv.edu.in