Journal of Information Systems Engineering and Management
2025, 10(11s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A New Approach to Natural Language Query Search using Frequency Analysis Techniques in Cloud Computing

Dr. Ninad More[1], Dr. Swapnali Makdey[2], Dr. Kirti Wanjale[3], Dr. Puja Padiya[4] , Dr. Nilesh Marathe[5] , Dr. Kranti Vithal Ghag[6]

[1]*Assistant Professor, Department of Computer Science and Engineering, Xavier Institute of Engineering, Mumbai University, Mumbai, India. ninad.m@xavier.ac.in*
[2]*Assistant Professor, Department of Artificial Intelligence and Data Science, Fr. Conceicao Rodrigues College of Engineering, Mumbai, Maharashtra, India. swapnalimakdey@gmail.com*
[3]*Professor, Computer Engineering Department, Vishwakarma Institute of Technology, Pune, India. kirti.wanjale@vit.edu*
[4]*Assistant Professor, Department of Computer Engineering, Ramrao Adik Institute of Technology, D Y Patil Deemed to be University, Nerul, Navi Mumbai, India. puja.padiya05@gmail.com*
[5]*Associate Professor, Department of Computer Science and Engineering (Data Science), Dwarkadas J. Sanghvi College of Engineering, Vile-Parle, Mumbai, India. nilmarathe2307@gmail.com*
[6]*Associate Professor, Department of Computer Engineering, SVKMs Dwarkadas J Sanghvi College of Engineering, Mumbai, India. kranti.ghag@gmail.com*

ARTICLE INFO

Received: 15 Nov 2024
Revised: 27 Dec 2024
Accepted: 15 Jan 2025

ABSTRACT

Data outsourcing is the preferred act by owners of cloud data because of ease in maintenance. Data confidentiality of this outsourced sensitive data is a major task. Applying cryptographic techniques to outsourced data is the most secure way to achieve confidentiality. Searchable encryption techniques help in searching on encrypted data without actually decrypting it. These techniques limit the usefulness of data in the sense that searching encrypted data is difficult. With the increase in the volume of data, increases the size of indexing structure. This makes it more difficult to design cipher text based search scheme which facilitates reliable, memory efficient, fast retrieval on a huge volume of encrypted data. In this paper, keyword frequency based code method is presented to minimize the size of the index which makes fast retrieval of data inside a big data environment. The proposed system also supports secured, ranked retrieval using hash-based mapping structures. A hash-based technique efficiently retrieves the data using key-value pair data structure. The resulting system is able to handle queries which are written in natural language. Through extensive experiments using standard dataset, the performance of the system is validated. The system performance is evaluated and validated through extensive experimentation using standard dataset. The results show that very less space for index storage is utilized by the system proposed in the paper which enhances the performance on the ground of the retrieval time.

Keywords: Cloud security, Searchable encryption, Frequency codes, Natural language query.

1. Introduction:

The cloud computing services provided by the massive data centers of corporations such as Google, Microsoft, draw several new customers every day. Cloud computing is an extended view of computing in which users can store the data in remote manner on cloud servers and access programs of high-quality to facilitate from a pool of shared, programmable resources [1]. The cloud computing paradigm is "pay-per-use." In this paradigm, consumers only pay for the resources they use [2]. This innovative computing paradigm provides relief for storage administration, global data access from several geographical areas, with drastic avoidance in expenditures in regard to hardware as well as software, and prominent reduction in cost of employee maintenance [3]. Irrespective of advantages provided by cloud computing paradigm, there are various challenges arising like data portability, data migration, and security.

Due to the large file sizes stored in the cloud, relevance ranking is required to avoid returning duplicate results. Therefore, users of data can quickly find the most relevant information thanks to ranked search algorithms.

Data consumers need not peruse the entire archive [4].

Some techniques [5] are available for multiple keyword searches on plain text data, but the biggest issue is how to apply multiple keywords searcheson encrypted data files and retrieve relevant data. One of the most strict requirement while retrieving data from cloud server security is to maintain index privacy, keyword privacy, data privacy, retrieval privacy and much more. Using keyword dictionary expansion, the author of [6] provides a versatile multi-keyword query approach that significantly reduces the maintenance burden. The contribution of proposed Frequency Based Multikey word Dynamic Ranked Search (FBDMRS) scheme are listed below:

1) Proposed system uses keyword frequency based codes for secure index construction which significantly reduce the size of index. This is very important in pay-as-you use scenario as it saves lot of space required on cloud servers.

2) Proposed system explores handling of natural language query ranked search which is then can be converted into multiple keywords over outsourced encrypted data and strictly maintain data privacy requirement.

3) Through analysis of proposed system is given and experimentation is done using real world dataset which proves that propose system introduces low overhead on space, time, communication and computation cost.

4) It maintains data integrity of retrieved result.

2. Related Work:

Cloud server searches the requested trapdoor of the keyword on the encryptedindex and retrieve relevant data and send it back to the user. A probable solution for cryptographic storage facility for cloud data is suggested in [7]. It suggested searchable encryption scheme using symmetric key cryptography technique e.g. Encryption using the AES algorithm with a private key. The similar technique is used in many research projects. The bloom filter technology is suggested in [8]. This technique is used to verify if the keyword part of an index. Song et al. A deterministic encryption technique to encrypt keywords for security is proposed in [9]. In [10] author advised a new technique on secure ranked query which uses file length and frequency of term. An asymmetric encryption scheme to construct searchable encryptions is advised in [11]. In this method, data owner encrypts index with public key and at the same time, authorized user's uses its companion private key to carry out searches. This assumes that key is securely shared among data users using standard key distribution techniques. Similar concepts are discussed in [12], [13]. All discussed techniques highlight on queries with single keyword.

Many people suggest techniques to handle multi-keyword queries to enhance search functionalities. One such technique is suggested in [15] which uses bilinear map. Fuzzy keyword search technique is discussed in [14]. To facilitate disjunctive and conjunctive search, predicate-based search queries [16, 17] are described. Some strategies are provided in [18], [19] that discuss ranked searching techniques employing order maintaining algorithms. These techniques only support single keyword searches. [20] Propose a privacy-preserving multi-keyword strategy. [20] Propose a secure multi-keyword search system that employs a ranking function based on a similarity index. [19] Discuss a safe multi-keyword search approach that uses the local sensitive hash (LSH) function to generate clusters. Discusses on the multi-user environment method can be seen in some literature. All papers are transformed into fixed length words and indexed individually using the approach [9]. [8] Propose a method for creating an additional index for each file. [24] Propose searchable encryption based on inverted indexing. [22] Discuss the T-Set data structure for keyword tuple names. [20] Propose a dynamic searchable approach based on this structure. Literature also presents preliminary work on the suggested system.

3. The Proposed System:

As demonstrated in figure 1, the system comprises of three fundamental components named data consumer, data owner and cloud server. The proposed techniques comprises of following modules.

3.1 Frequency based keyword code generation scheme (FBKCS)

3.2 Natural language query processing (NLQP)

3.3 Index construction and searching of FBDMRS Scheme

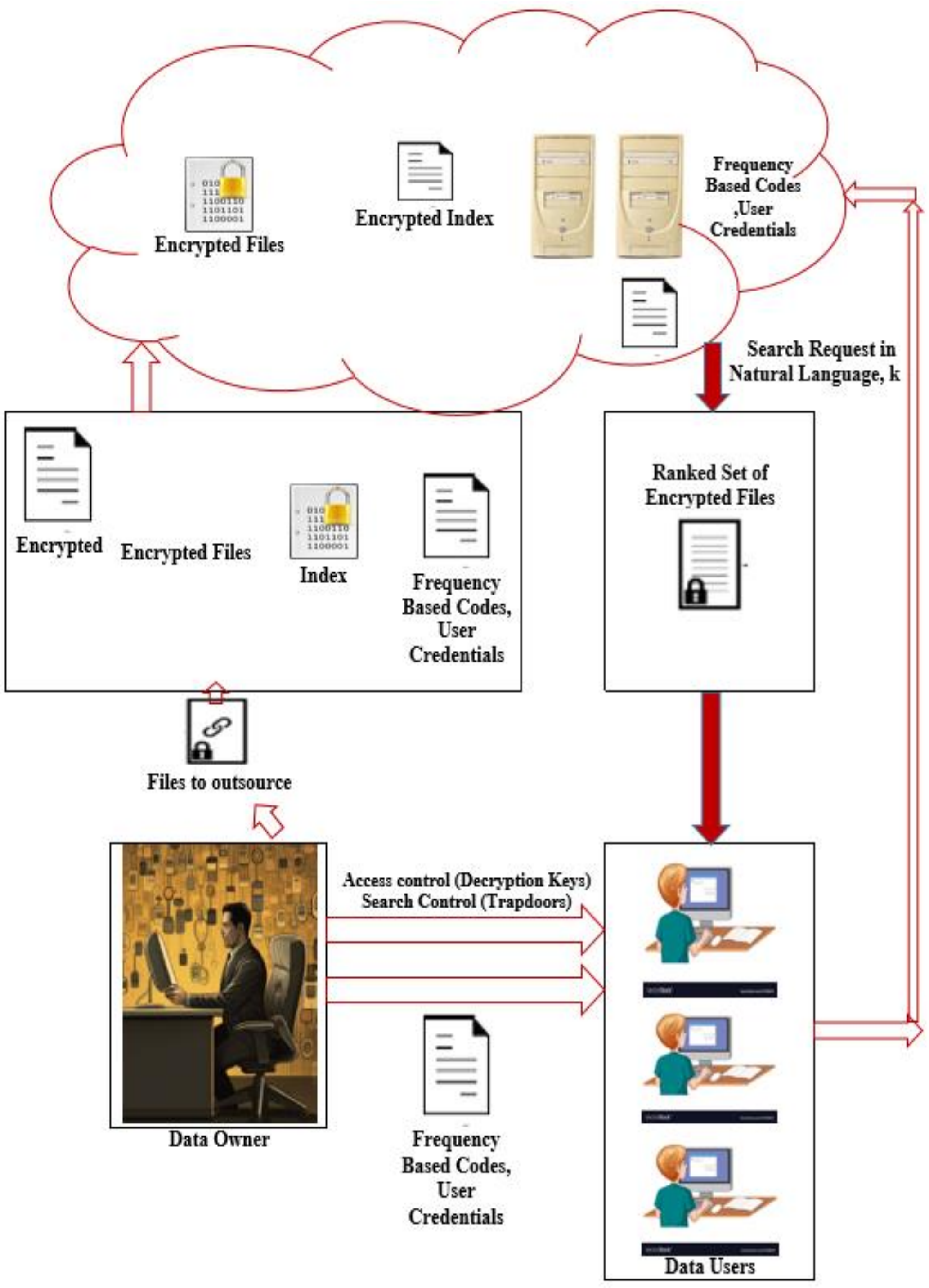

Encrypted Files
Encrypted Index
Frequency Based Codes ,User Credentials
Search Request in Natural Language, k
Encrypted
Encrypted Files
Index
Frequency Based Codes, User Credentials
Ranked Set of Encrypted Files
Files to outsource
Access control (Decryption Keys)
Search Control (Trapdoors)
Data Owner
Frequency Based Codes, User Credentials
Data Users

Figure.1 The architecture of frequency based dynamic multi-key word ranked search.

1. U_{cred}: User credentials
2. T_{wu}: Trapdoor of requested keyword
3. $freq_i$: Frequency of ith keyword

3.1 Frequency Based Keyword Code Generation Scheme (FBKCS):

F requency based codesfor all the keywords present in the document collection are generated first. Build-Tree and Build-code algorithms are given in first and second Algorithm.

Algorithm 1: Build-Tree (W)

```
Input: Keywords wi ε W,freqi ε FREQ
Output: Frequency tree
for each keyword wi ε W do
Qi=Freq(wi)
end for
n=|W|
for i=1 to n-1 do
Construct a new node new1 for wi
new1.left=x1=remove. min(Q)
new1.right=y1=remove. min(Q)
new1.freq=x1.freq+y1.freq
insert (Q, new1)
end for
return (remove. min(Q)) return root of the tree
```

Algorithm 2: Build-code (T)

```
Input: Root of the tree constructed in Algorithm 1 root=Build-Tree (W)
Output: Frequency codes of keywords wi ε W
if (root.left) then
code[i]=0
3.    i = i+1
call Buildcode (root.left)
end if
if (root.right) then
code[i]=1
8.    i = i+1
call Buildcode(root.right)
end if
if root has reached the leaf node then
FBC(wi)=code
end if
return to main function
```

3.2 NLQP (Natural Language Query Processing of User's Request:

The data users of outsourced cloud data are of all categories. Some users are expert while others may be naive. It becomes easy for naive users to write a query in the natural language like English. If we consider the scenario of Health-care data, there are multiple users of this data from sophisticated doctors to insurance agents. Insurance agents might not be aware of various formats and writing skills. Hence it becomes difficult for them to use the system. System usage will greatly improve if it is less restrictive. We try to provide a solution where users can write a query in natural language, we considered here English.Queries are preprocessed and significant keywords are extracted. Query processing is done through three important stages. If there are any punctuation marks or spacing that are removed first. Frequently occurring words are called as stop words. These stop words do not add any meaning to the query and hence can be removed. We have considered latest stop word list which comprises of around 429 words. Next phase is suffix stripping, we have used standard Porters algorithm to remove the suffixes. There is any duplication that is removed in last phase. Finally, we get the important list of keywords from the query.

3.3 Index Construction and Searching of FBDMRS Scheme:

3.3.1 Index construction:

The OPSE score computation method is utilised in the building of the inverted index. The index construction approach presented in the paper is closely related to recent work [19], while we are primarily concerned with memory conservation and search effectiveness..

Basic scheme: Let k_1, l_1, o_1, p_1 be parameters incorporating security that will be implemented in key generation. Let δ be a semantically secure symmetric encryption algorithm δ: $(0, 1)^{l1} \times (0, 1)^{r1} \rightarrow (0, 1)^{r1}$. Let v_1 be the maximum file number having some keyword $w_i \varepsilon W$ for i=1,.....,m_1, i.e $v_1=max(N_i)$. Let f_1 be a pseudorandom function and π be a collision resistant hash function.

In the setup phase

During the setup phase data owner creates random keys $(x_1,y_1\} \overset{R}{\leftarrow} (0,1)^{k1}$ and

$Z_1 \overset{R}{\leftarrow} (0,1)^{l1}$ and outputs set of secret keys $k_1=\{x_1,y_1,z_1\}$.

3.3.2 Searching:

1. Using Hash based Technique:

Data User submits a query in natural language. This query is pre-processed as discussed to extract significant query wordsfrom it. Data users generate the trapdoor by using key and frequency code shared by data owner

$$T_{wu}=(FBC(w_u),k,f_{y1}(w_u),E_{fy1}(U_{cred})) \text{ by calling trapdoorGen}(w_u).$$

After receiving the trapdoor T_{wu} the server call search Index (I_e, T_{wu}). It first decrypts user credentials and then checks for its authenticity. If he is the authorized user then it computes the hash of *FBC* (w_u) and locates the matching index list. It then uses $f_{y1}(w_u)$ for decryption of entries andsend back corresponding k encrypted files along with associated encrypted relevance scores and digest value. Data users can retrieve ranked result by decrypting the relevance score using key z_1.

Data users use message digest algorithm to compute the digest value ofthe retrieved file and then compare it with the transmitted value. If both values matches he concludes that file is not tampered in the transmission or by cloud server and thus data integrity is checked. Details are given in Algorithm 3.

Search algorithm is composed of three important parts (Eq.1, Eq.2 and Eq.3)

Checking authenticity:

$$\text{auchk}(U_{credi}, U_{credu}) = \{ 1, \text{if autheticated user and}$$
$$0 \text{ otherwise}$$

where $i = 0$ to l, l: size of user credential table on server

Searching top k files:

$$\mathrm{hmatch}\left(\mathrm{FBC}(w_u), I_e\left(\mathrm{FBC}(w_i)\right)\right) = 1, \text{ if match found}$$
$$= 0 \text{ otherwise}$$

Checking integrity of retrieved result:

$$\mathrm{intchk}(\mathrm{digest}(f_{iu}), \mathrm{md}_{iu}) = 1, \text{ if file not tempered}$$
$$= 0 \text{ otherwise}$$

The discussed technique noticeably fulfills the security guarantee of SSE. Data user sends the value of top k documents he wants. Since the files in the index are arranged by using OPSE scores, the server can easily identify first top $k$files. Server will not grasp any relevance score value in this manner, but it will realise that prominently the significant contribution is by the fikes which are retrieved than non-retrieved files. This may expose more sensitive information than the search and access pattern.

2. Using frequency based tree:

To improve search efficacy, we suggest a frequency code tree built using a finite collection of frequency codes. The major goal of this structure is to allow any frequency-based codes with a common prefix to share nodes. The root is set to the value NULL. From root to Leafs, symbols in the frequency-based code of the requested keyword are searched. The depth-first search technique is used to search all of the corresponding documents. When the search request reaches the server from a data user, it begins searching for files using algorithm 3 and provides the top k documents together with an encrypted score and message digest, as indicated in equation 4.

$$E_{fy}((id(f_{u1})||\, E_z(S_{u1})||MD\ (f_{u1}))\ \dots\dots\dots\dots.\ ((id(f_{u1})||\, E_z(S_{u1})||MD\ (f_{u1})$$

Eq.4

No information about the requested keyword is exposed when using this scheme. Because frequency-based codes share a common route, efficiency is increased. The encrypted file identifiers, as well as the score and digest, are kept at the tree's leaf node, as shown in Fig.2. Dotted leaf nodes have the same structure as the down pointer of the V7 node. Algorithm 4 contains more information. Data users can decrypt files, scores, and utilize digests to ensure the file's integrity. The search cost for each request is O (l), where l represents code length, and hence independent of file size and requested keyword.

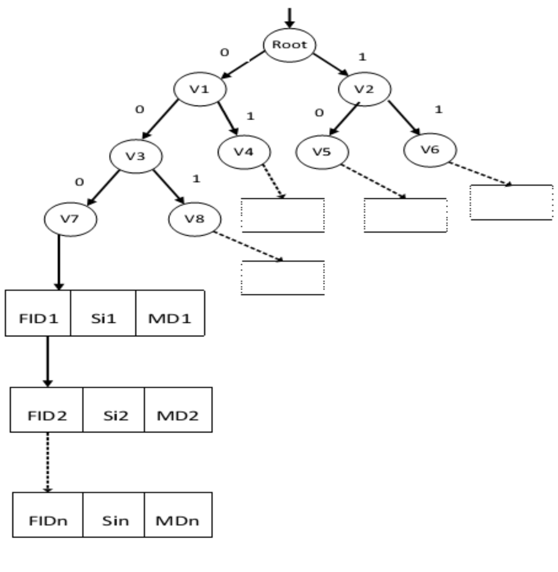
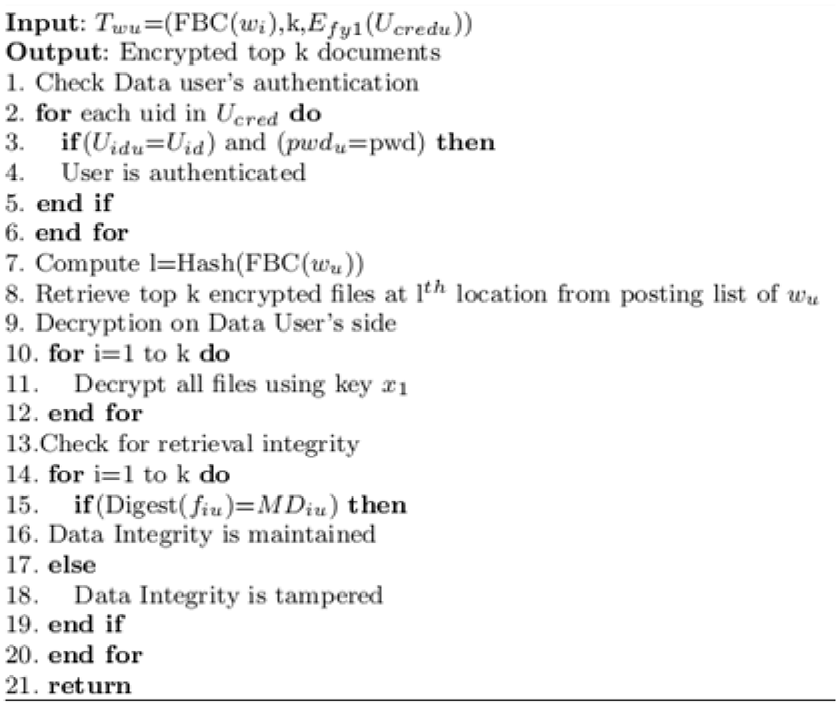


Figure. 2 Searching using frequency based code tree

Algorithm 3: Search-H-Index (Ie, Twu)

```
Input: $T_{wu}$=(FBC($w_u$),k,$E_{fy1}(U_{credu})$)
Output: Encrypted top k documents
1. Check Data user's authentication
2. for each uid in $U_{cred}$ do
3.   if($U_{idu}$=$U_{id}$) and ($pwd_u$=pwd) then
4.     User is authenticated
5.   end if
6. end for
7. for i=1 to $|FBC(w_u)|$ do
8.   set currentnode as root of $G_W$
9.   if $i^th$ bit of $|FBC(w_u)|$ is '0' then
10.     currentnode=currentnode→ left
11.   end if
12.   if $i^th$ bit of $|FBC(w_u)|$ is '1' then
13.     currentnode=currentnode→ right
14.   end if
15. end for
16. if currentnode is leaf node then
17.   for j=1 to k do
18.     retrieve jth node pointed by down pointer and append it to resultset
19.   end for
20.end if
21. return result set
```



```
Input: $T_{wu}$=(FBC($w_i$),k,$E_{fy1}(U_{credu})$)
Output: Encrypted top k documents
1. Check Data user's authentication
2. for each uid in $U_{cred}$ do
3.   if($U_{idu}$=$U_{id}$) and ($pwd_u$=pwd) then
4.   User is authenticated
5. end if
6. end for
7. Compute l=Hash(FBC($w_u$))
8. Retrieve top k encrypted files at $l^{th}$ location from posting list of $w_u$
9. Decryption on Data User's side
10. for i=1 to k do
11.   Decrypt all files using key $x_1$
12. end for
13.Check for retrieval integrity
14. for i=1 to k do
15.   if(Digest($f_{iu}$)=$MD_{iu}$) then
16. Data Integrity is maintained
17. else
18.   Data Integrity is tampered
19. end if
20. end for
21. return
```

Algorithm 4: Search FBT (T_{wu}, k)

```
Input: $T_{wu}$=(FBC($w_u$),k,$E_{fy1}(U_{credu})$)
Output: Encrypted top k documents
1. Check Data user's authentication
2. for each uid in $U_{cred}$ do
3.    if($U_{idu}$=$U_{id}$) and ($pwd_u$=pwd) then
4.       User is authenticated
5.    end if
6. end for
7. for i=1 to $|FBC(w_u)|$ do
8.   set currentnode as root of $G_W$
9.   if $i^th$ bit of $|FBC(w_u)|$ is '0' then
10.      currentnode=currentnode→ left
11.   end if
12.   if $i^th$ bit of $|FBC(w_u)|$ is '1' then
13.      currentnode=currentnode→ right
14.   end if
15. end for
16. if currentnode is leaf node then
17.    for j=1 to k do
18.       retrieve jth node pointed by down pointer and append it to resultset
19.   end for
20.end if
21. return result set
```

There are many times when the data owner updates document collection. All these updates result in the change of frequency based codes. The data owner is responsible for sharing these updated codes. Table 1 showcases keywords with three categories (classes) mentioned with reference to frequency distribution in the document collection.

Table 1: Frequency based codes for some keywords from document collection

Keyword	Frequency	Code
Routing	1259	0
IAB	633	10
service	318	110
Route	79	11100
Objects	10	11101000
encoding	5	111010010
UDP	5	111010011
vector	20	1110101
statistics	20	1110110

4. Performance Analysis:

Under the Windows 10 operating system, the suggested system is implemented utilizing Java and its cryptographic libraries.

The system is put to test using the dataset for comment (RFC) [30]. The files are chosen at random from the corpus. The processor is a 2.2 GHz Intel CORE i3 CPU. The index construction, search, and update are all part of the test. We compare our scheme to those of [19] and [31]. It is worth noting that our FBDMRS strategy provides great search performance by precisely calculating keyword location in the index using the hash technique. As a result, top-k search precision is very high.

4.1 Index Construction:

Index generation time presents highly sensitivity towards the document number and keywords included with the input corpus. According to Figure 3, the amount of time needed to construct the index scales almost linearly with collection size. Index creation is a time greedy procedure that should be taken into account. It stands out because it is an exceptional task.

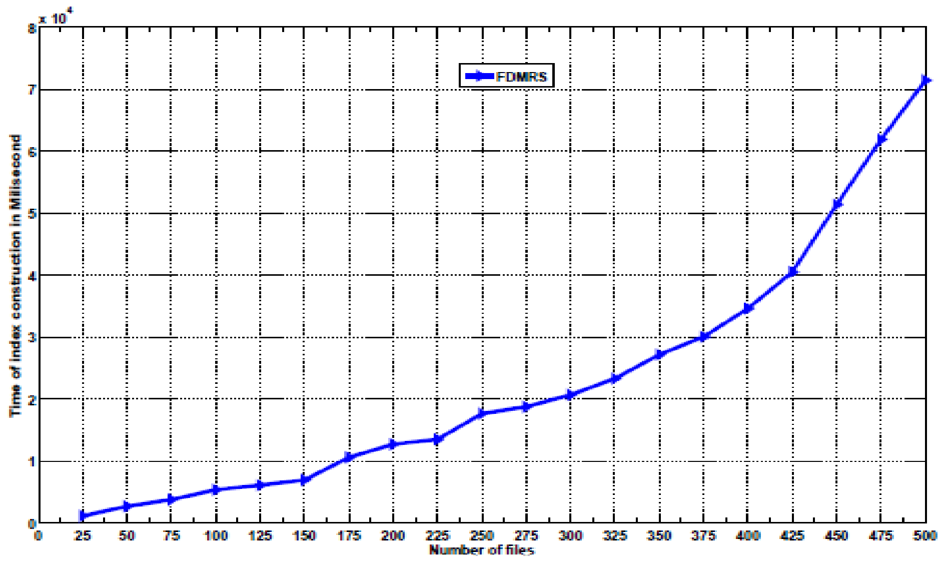


Figure. 3 Time Required to Build the Index

4.2 Search efficiency

We have compared our scheme with Wang et al. [19]. Graph shows time requirefor top-k retrieval. Fig. 4 shows that our scheme requires less time for searching.It is clear from the graph that our scheme is better.

In [19] author used message digest and then encrypted it for searching. The length of message digest and length of frequency codes differs with substantial amount. Hence searching is faster with frequency codes. Even though frequency codes are traceable sometimes, but if they are encrypted it is difficult for the attacker to break it.

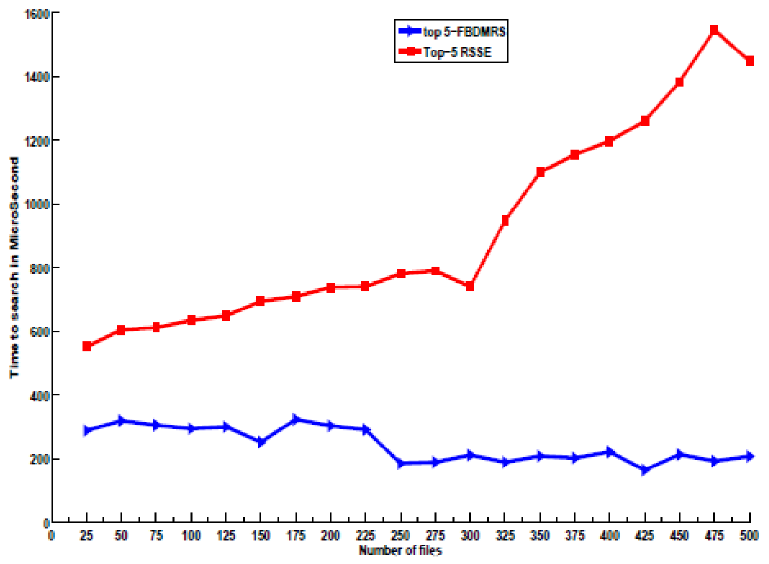


Figure. 4 Search comparison between RSSE and FDBMRS

5. Conclusion and Future Scope:

The suggested method discusses a secure, prioritized, and efficient search mechanism for encrypted data that has been outsourced. We build the index with frequency-based coding, which greatly reduces the index's size. The index size necessary for regular vocabulary is substantially bigger than the index size required for frequency codes. Furthermore, search using natural language queries can be performed to make the system more user-friendly. The suggested system processes natural language queries and extracts relevant keywords from them. This system also provides authentication and retrieval integrity for data users. The experimental findings demonstrate the effectiveness of the proposed system.

There are still numerous difficult difficulties in the field of searchable encryption. When a query is in the form of a phrase, significant work can be done to carry out the search. The multi-user system also presents other obstacles. In a multiuser environment, all users keep a key for trapdoor generation that the data owner shares. The revocation of these keys is a significant difficulty.

Artificial intelligence can be used to improve search results. This system's future development will be to manage more accurate findings utilizing artificial intelligence.

References:

[1] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on, pp. 253–262, IEEE, (2010).

[2] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on, pp. 253–262, IEEE, (2010).

[3] R. Buyya, J. Broberg, and A. M. Goscinski, Cloud computing: Principles and paradigms, vol. 87. John Wiley & Sons, 2010.

[4] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "Above the clouds: A berkeley view of cloud computing," Dept. Electri- cal Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, vol. 28, no. 13, p. 2009, 2009.

[5] A. Singhal, "Modern information retrieval: A brief overview," IEEE Data Eng. Bull., vol. 24, no. 4, pp. 35–43, 2001.

[6] I. H. Witten, A. Moffat, and T. C. Bell, Managing gigabytes: compressing and indexing documents and images. Morgan Kaufmann, 1999.

[7] R. Li, Z. Xu, W. Kang, K. C. Yow, and C.-Z. Xu, "Efficient multi-keyword ranked query over encrypted data in cloud computing," Future Generation Computer Systems, vol. 30, pp. 179–190, 2014.

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

[9] E.-J. Goh et al., "Secure indexes.," IACR Cryptology ePrint Archive, vol. 2003, p. 216, 2003.

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

[11] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Verifiable privacy- preserving multi-keyword text search in the cloud supporting similarity-based ranking," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 11, pp. 3025–3035, 2014.

[12] Q. Liu, G. Wang, and J. Wu, "An efficient privacy preserving keyword search scheme in cloud computing," in Computational Science and Engineering, 2009. CSE'09. Inter- national Conference on, vol. 2, pp. 715–720, IEEE, 2009.

[13] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency prop- erties, relation to anonymous ibe, and extensions," in Annual International Cryptology Conference, pp. 205–222, Springer, 2005.

[14] P. Allirani, R. Jain, S. Shankar Prasd, K. H. Wanjale, A. Amudha and A. Faiz, "Real-Time Depth Map Upsampling for High-Quality Stereoscopic Video Display," 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kamand, India, 2024, pp. 1-5, doi: 10.1109/ICCCNT61001.2024.10725345

[15] 15.D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in International Workshop on Information Security Applications, pp. 73–86, Springer, 2004.

[16] J. Katz, A. Sahai, and B. Waters, “Predicate encryption supporting disjunctions, poly- nomial equations, and inner products,” in Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 146–162, Springer, 2008.

[17] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, “Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption,” in Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 62–91, Springer, 2010.

[18] K. Wanjale, A. A. Deshmukh, J. C. Vanikar, A. P. Adsul and S. P. Bendale, "Detecting Human Eye Blinks through OpenCV," 2024 IEEE 9th International Conference for Convergence in Technology (I2CT), Pune, India, 2024, pp. 1-5, doi: 10.1109/I2CT61223.2024.10543678

[19] 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 sys- tems, vol. 23, no. 8, pp. 1467–1479, 2012

[20] 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 sys- tems, vol. 25, no. 1,pp.222–233, 2014.