Novel Frequency Based Natural Language Query Search in Cloud Computing

Bharati P. Vasgi[1], Girija G. Chiddarwar[2], S. M. Jaybhaye[3]
[1,2]Marathwada Mitra Mandal's COE, Pune.
[3]Vishwakarma Institute of Technology, Pune, India.

Abstract:
Cloud data owners prefer to outsource 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 usability of data in the sense that it is difficult to search on encrypted data. As the volume of data increases the size of indexing structure increases. 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 results show that the proposed system requires very less space for index storage and hence also improves the retrieval time.

Keywords: Cloud security Searchable encryption Frequency codes Natural language query
DOI: 10.24297/j.cims.2023.5.1

1. Introduction

The cloud computing services presented by large-scale data centers of companies like Google, Microsoft, or Amazon attract every day many new clients. Cloud computing is an elongated vision of computing as a utility, in which users can remotely store their data on cloud servers and avail on-demand high-class applications and facilities from the shared pool of configurable computing re- sources [1]. Cloud computing is a 'pay-as-per-use' model. In this model, users pay as per resources utilized [2]. The benefits brought by this novel computing model are relief of the load for storage administration, worldwide data access with different geographical locations, and avoidance of capital cost on hardware, software, and personnel maintenance [3].

Irrespective of advantages provided by cloud computing paradigm, there are various challenges arising like data portability, data migration, and security.

Big size of files on the cloud server needs relevance ranking rather than re- turning undifferentiated results. Thus ranked searching techniques helps data users to retrieve top relevant information in less time. Data users need not have to scan the entire collection [4].
Some techniques [5] are available for multiple keyword searches on plain text data, but the biggest issue is how to apply multiple keywords searches on 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. In [6] author proposes a flexible multi-keyword query scheme which noticeably decreases the maintenance overhead by using the keyword dictionary expansion.

The contribution of proposed Frequency Based Multikeyword 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 omputation cost.
4. It maintains data integrity of retrieved result.

2. Related Work

Cloud server searches the requested trapdoor of the keyword on the encrypted index 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. AES under a unique 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 is included in 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 his public key and authorized users 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].

Predicate-based search queries [16], [17] are presented to support disjunctive and conjunctive search. There are some methods presented in [18], [19] which dis- cuss about ranked searching technique using order preserving method. These techniques support only single keyword based searches. Privacy preserving multi-keyword scheme is suggested in [20]. A secure multi-keyword search scheme that uses ranking function based on similarity index is put forward in [21]. A secure multi keyword search method which utilized local sensitive hash (LSH) function to form the clusters is discussed in [22]. The multi-user environment technique is discussed in [23]. In the method [9], all documents are converted into fix length words and are indexed individually. A scheme to generate an additional index for every file is proposed in [8]. Searchable encryption which uses the inverted indexing is proposed in [24]. A data structure for keyword tuple names T-Set is discussed in [25]. Based on this structure, dynamic searchable technique is proposed in [26]. Initial work of the proposed system is presented in [27]. Extension to this work using genetic algorithm is presented in [32]. More recent work on outsourced data security is discussed in [33], [34] and [35].

3. The Proposed System

The system is composed of three basic entities: data owner, cloud server and data user as shown in Fig. 1

The proposed technique comprises of following modules

- Frequency based keyword code generation scheme (FBKCS)
- Natural language query processing (NLQP)

Index construction and searching of FBDMRS Scheme Following are the notations used

- *C:* Set of files to be outsourced, $C = \{F_1, F_2, \cdots, F_n\}$
- *W:* Set of distinct keywords derived from the file collection C, $W = \{w_1, w_2, \cdots, w_{m1}\}$
- $_{m1}$: Total number of keywords
- $f(w_i)$: Set of identifiers of files in C that contain keyword w_i
- $FBC(w_i)$: Frequency based code of w_i
- *N:* Total number of Files to be outsourced
- h_i: Hash value of w_i i.e $\pi(w_i)$
- $id(F_j)$: Identifier of file F_j
- W_u: Keyword of search request
- I_p: Plaintext Index structure
- $F_{d,t}$: Term frequency TF of term t in file f_d
- f_t: Number of files that contain term t
- $|fd|$: Number of indexed terms present in file f_d
- π: A collision resistant hash function e.g SHA-1
- k: Top k documents required by data user
- E: Encrypted document collection

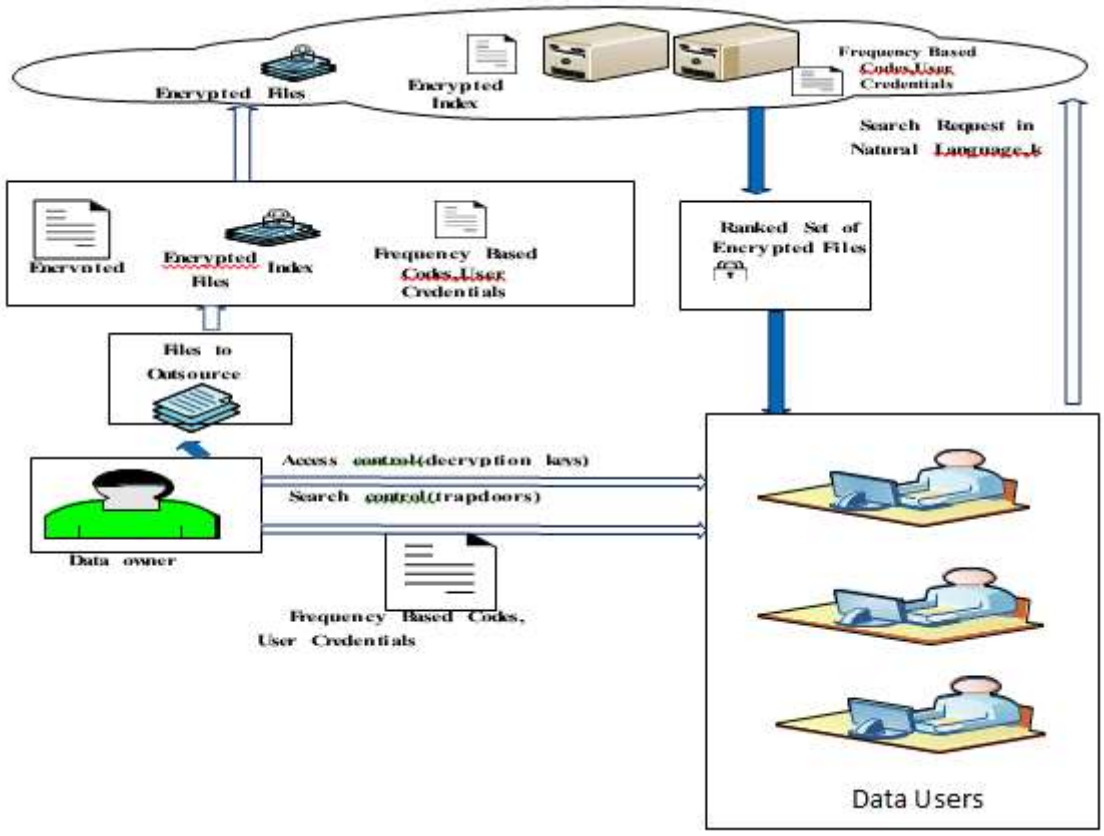

Fig. 1 The architecture of frequency based dynamic multikeyword ranked search

- U_{cred}: User credentials
- T_{wu}: Trapdoor of requested keyword
- $freq_i$:Frequency of i^{th} keyword

3.1 Frequency based keyword code generation scheme(FBKCS)

Frequency based codes for all the keywords present in the document collection are generated first. More details are dis- cussed in [28]. BuildTree and Buildcode algorithms are given in Algorithm 1 and Algorithm 2.

Algorithm 1 BuildFTree(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 Buildcode(T)
Input:Root of the tree constructed in Algorithm 1 root=BuildTree(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 is leaf node then
FBC(wi)=code
end if
return

3.2 NLQP (Natural Language Query Processing) of users 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

In the construction of the inverted index, OPSE score calculation methods is adopted [29]. Note that the index construction scheme that we discuss here is tightly pertained to recent work [19], though we focus on memory saving and search efficiency.

Basic scheme: Let k_1, l_1, o_1, p_1 be security parameters that will be used in key generation. Let δ be a semantically secure symmetric encryption algo- rithm δ: $(0, 1)^{\prime 1} \times (0, 1)^{\prime 1} \rightarrow (0, 1)^{\prime 1}$. Let v_1 be the maximum number of files 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 R

During the setup phase data owner creates random keys $(x_1,y_1\} \leftarrow (0,1)^{k1}$ and R $Z_1 \leftarrow (0,1)^{l1}$ 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 words from it.

Data users generate the trapdoor by using key and frequency code shared by data owner

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

After receiving the trapdoor T_{wu} the server call searchIndex (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 list of index. It then uses $f_{y1}(w_u)$ to decrypt the entries and send 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 of the 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

$$auchk(U_{credi}, U_{credu}) = \begin{cases} 1, & \text{if } authenticated\ user \\ 0 & \text{otherwise} \end{cases}$$

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

– Searching top-k files

$$hmatch(FBC(w_u), I_e(FBC(w_i))) = \begin{cases} 1, & \text{if } match\,found \\ 0 & \text{otherwise} \end{cases}$$

– Checking integrity of retrieved result

$$intchk(digest(f_{iu}), md_{iu}) = \begin{cases} 1, & \text{if } file\ not\ tampered \\ 0 & \text{otherwise} \end{cases}$$

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. Take note of that this way server will not understand any value of relevance scores, but it recognizes the retrieved files are more important than non-retrieved. This can leak more important information rather than search and access pattern.

2.Using frequency based tree

To improve search effectiveness, we propose frequency code tree which is con- structed using the finite set of frequency codes. The main intention behind this construction is that all the frequency based codes, sharing common prefix may have common nodes. The root is set to NULL value. Symbols in the frequency based code of requested keyword are searched from root to Leafs. All the corresponding documents are searched using depth-first-search technique. After receiving the search request from data user server starts searching for the files using algorithm 3 and returns top k documents along with encrypted score and message digest as shown in equation 4.

$$E_{fy}\left(\left(id(f_{u1})||E_z(S_{u1})||MD(f_{u1})\right)\cdots\cdots\left(\left(id(f_{u1})||E_z(S_{u1})||MD(f_{u1})\right)\right) \quad \text{Eq.4}$$

Using this scheme no information about the requested keyword is leaked. Since the frequency based codes share common path it improves the efficiency. The encrypted file identifiers along with score and digest are stored at the leaf node of the tree as shown in Fig.2. Dotted leaf nodes indicate the same structure as that of to the down pointer of V_7 node. Details are given in Algorithm 4. Data users can decrypt the files, scores and make use of digest to check the integrity of the file. For each request, search cost is O(l) where l is the length of the code and hence it is independent of the size of the file and requested keyword.

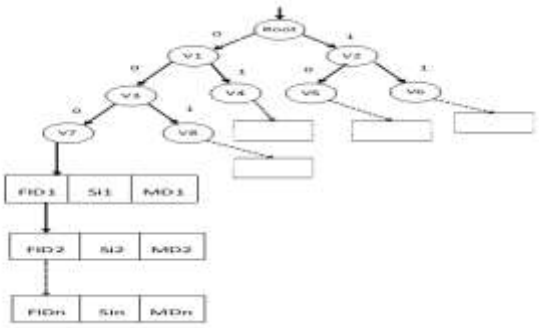

Fig. 2 Searching using frequency based code tree

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

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(Twu, 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. In Table 1 there are three classes of keywords as per their frequency distribution in the 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

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

4. Performance Analysis

The proposed system is implemented using Java and its cryptographic libraries under Windows 10 operating system.

The system is tested using request for comment (RFC) [30] dataset. The files are selected randomly from the corpus. The processor is Intel CORE i3 CPU running at 2.2 GHz. The test includes the index construction, search, and update. We compare our scheme with work proposed in [19] and [31]. Note that our scheme FBDMRS achieves high search efficiency through exact calculation of keywords location in the index using the hash technique. Hence top-k search precision is very accurate.

4.1 Index Construction

The time requires to build the index importantly depends upon the number of documents and keywords present in the input corpus. Fig. 3 shows that the time required to construct the index is almost linear with respect to the number of documents in the collection. It is noted that time required for index construction is greater than other operations. It is noteworthy as it is a onetime task.

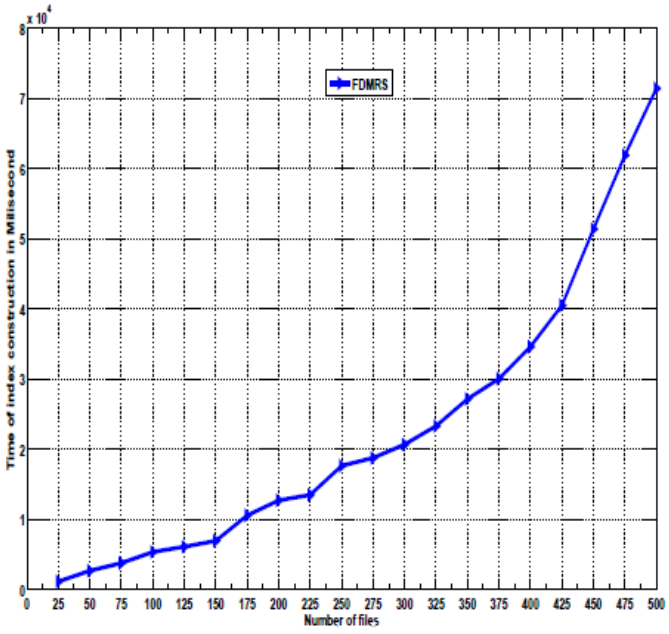

Fig. 3 Time Required to Build the Index

4.2 Search efficiency

We have compared our scheme with Wang et al. [19]. Graph shows time require for 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.

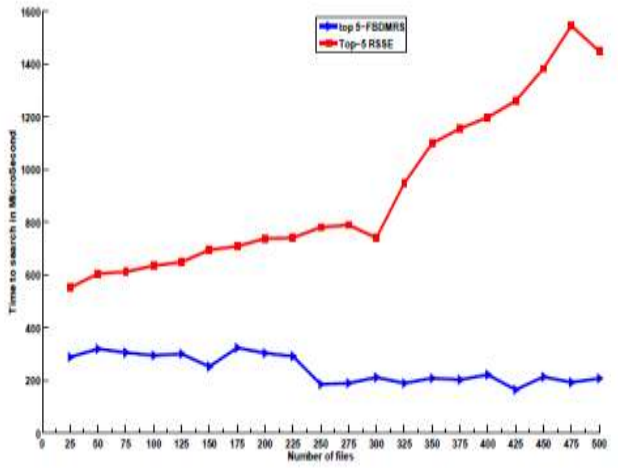

Fig. 4 Search comparison between RSSE and FDBMRS

5. Conclusion and Future Scope

The proposed system discusses secure, ranked and efficient search technique on outsourced encrypted data. We construct the index using frequency based codes which reduce the size of index significantly. The size required for index using normal vocabulary is much larger than index constructed using frequency codes. In addition to this, search using the query in natural language can be carried out to make the system more user-friendly. Natural language query is processed by the proposed system and important keywords are extracted from the query. This system also supports data users authentication and retrieval integrity. Experimental results prove the efficiency of the proposed system.

There are still many challenging problems in the area of searchable encryption. Significant work can be done to carry out the search when a query is in the form of the phrase. There are also many challenges in the multi-user system. In the multiuser environment, all the users keep a key for trapdoor generation which is shared by the data owner. Revocation of these keys is a major challenge.

Artificial Intelligence can be used for search optimization. The future work of this system is to handle more accurate results using 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. R. Buyya, J. Broberg, and A. M. Goscinski, Cloud computing: Principles and paradigms, vol. 87. John Wiley & Sons, 2010.
3. 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.
4. A. Singhal, "Modern information retrieval: A brief overview," IEEE Data Eng. Bull., vol. 24, no. 4, pp. 35–43, 2001.

5. I. H. Witten, A. Moffat, and T. C. Bell, Managing gigabytes: compressing and indexing documents and images. Morgan Kaufmann, 1999.
6. 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.
7. S. Kamara and K. Lauter, "Cryptographic cloud storage," in International Conference on Financial Cryptography and Data Security, pp. 136–149, Springer, 2010.
8. E.-J. Goh et al., "Secure indexes.," IACR Cryptology ePrint Archive, vol. 2003, p. 216, 2003.
9. 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.
10. 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.
11. 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. International Conference on, vol. 2, pp. 715–720, IEEE, 2009.
12. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee,
13. G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," in Annual International Cryptology Conference, pp. 205–222, Springer, 2005.
14. J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in INFOCOM, 2010 Proceedings IEEE, pp. 1–5, IEEE, 2010.
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, polynomial 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. A. Swaminathan, Y. Mao, G.-M. Su, H. Gou, A. L. Varna, S. He, M. Wu, and D. W. Oard, "Confidentiality-preserving rank-ordered search," in Proceedings of the 2007 ACM workshop on Storage security and survivability, pp. 7–12, ACM, 2007
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 systems, 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 systems, vol. 25, no. 1, pp. 222–233, 2014.
21. W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," in Pro- ceedings of the 8th ACM SIGSAC symposium on Information, computer and commu- nications security, pp. 71–82, ACM, 2013.
22. C. Orencik, M. Kantarcioglu, and E. Savas, "A practical and secure multi-keyword search method over encrypted cloud data," in Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on, pp. 390–397, IEEE, 2013
23. W. Zhang, S. Xiao, Y. Lin, T. Zhou, and S. Zhou, "Secure ranked multi-keyword search for multiple data owners in cloud computing," in Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on, pp. 276–286, IEEE, 2014.
24. S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in Proceedings of the 2012 ACM conference on Computer and communications security, pp. 965–976, ACM, 2012.
25. D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Ro¸su, and M. Steiner, "Highly- scalable searchable symmetric encryption with support for boolean queries," in Advances in Cryptology–CRYPTO 2013, pp. 353–373, Springer, 2013.
26. D. Cash, J. Jaeger, S. Jarecki, C. S. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner, "Dynamic searchable encryption in very-large databases: Data structures and implementation.," in NDSS, vol. 14, pp. 23–26, Citeseer, 2014
27. B. Vasgi and U. Kulkarni, "A secure and effective retrieval using hash based mapping structure over encrypted cloud data," International Journal of Electrical Electronics and Computer Science Engineering, vol. 4, no. 4, pp. 65–74, 2017.

28. B. Vasgi and U. Kulkarni, "Secure sharing and retrieval of personal health records in outsourced environment," Journal of Applied Science and Computations, vol. 5, no. 7, pp. 14–20, 2018.
29. A. Boldyreva, N. Chenette, Y. Lee, and A. Oneill, "Order-preserving symmetric encryption," in Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 224–241, Springer, 2009.
30. RFC, "Request for comments database," http://www.ietf.org/ rfc.html, 2012.
31. Z. Xia, X. Wang, X. Sun, and Q. 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.
32. B. Vasgi and U. Kulkarni, "Secure retrieval in outsourced environment using genetic algorithm," Proceedings - 2019 5th International Conference on Computing, Communication Control and Automation, ICCUBEA 2019, 2019.
33. Shaon, Fahad, and Murat Kantarcioglu. "Sgx-ir: Secure information retrieval with trusted processors." In IFIP Annual Conference on Data and Applications Security and Privacy, pp. 367-387. Springer, Cham, 2020.
34. Ma, Wentao, Tongqing Zhou, Jiaohua Qin, Xuyu Xiang, Yun Tan, and Zhiping Cai. "A privacy-preserving content-based image retrieval method based on deep learning in cloud computing." Expert Systems with Applications (2022): 117508.
35. Zhu, Dan, Hui Zhu, Xiangyu Wang, Rongxing Lu, and Dengguo Feng. "An Accurate and Privacy-preserving Retrieval Scheme over Outsourced Medical Images." IEEE Transactions on Services Computing (2022).