Dynamic & Fuzzy Searchable Encryption System

Sunitha Buchade[#1], Prof. P. R. Devale[*2]

[#1]*Department of Information Technology, Bharati Vidyapeeth (Deemed to be) University College of Engineering, Pune, India.*

[#2]*Department of Information Technology, Bharati Vidyapeeth (Deemed to be) University College of Engineering, Pune, India.*

[1]puresoni06@gmail.com

[2]prakash.devale2012@gmail.com

Abstract*— To manage large volume of data, most of the organizations and data owners outsource their data to remote cloud storage servers. Since the remote servers are untrusted party, the data has to be encrypted to achieve security and privacy. But encryption and decryption causes communication overhead for many data utilization operations like searching and updating. To overcome this contradiction, searchable encryption was introduced. Searchable encryption (SE) is an ability of a server to search upon the ciphertext and retrieve the data without decrypting it. By searching on the ciphertext it protects the user's tactful data. In cloud computing, searching over encrypted data has led to the notion of searchable encryption (SE) scheme using keywords. Existing SE schemes retrieves relevant files containing the desired keywords but results are returned only when query keywords are complete and are exact match. That is there is no tolerance for minor spelling errors, impartial or incomplete fuzzy query keywords which on other hand is commonly made mistakes by users while searching. This affects user searching experience which might hinder usage of cloud facilities. We propose a SE scheme which achieves the search ability for incomplete fuzzy keywords. In our scheme, the keyword is split into individual characters and later encrypted along with corresponding file using deterministic AES encryption. For ease of data access control, we also encrypt the files and keywords with attribute based encryption. To achieve the security and privacy we use lookup table to store encrypted keywords for search operation. Moreover, this scheme can be dynamic to support file edit operation by data owner without the need for re encryption along with addition and deletion option. In this paper, we propose another trump card accessible encryption framework to help special case catchphrase inquiries which has a few profoundly attractive highlights..

Keywords*— searchable encryption, Fuzzy search, Attribute based encryption, top-k, AES Algorithm.

I. INTRODUCTION

Distributed storage gives supporters pervasive, dynamic, versatile and on-request stockpiling administration. While distributed storage brings more comfort and advantage than any time in recent memory, it likewise presents critical security and protection dangers to clients' information. To guarantee security of the outsourced information in the untrusted open cloud, information encryption is a compelling method to forestall inside/outside enemies from getting to the touchy data. In the interim, it is important to help information recovery work (without unscrambling) on scrambled information to encourage the information use. Such system is alluded to as searchable encryption (SE). To accomplish adaptable inquiry works, the idea of special case accessible encryption was proposed to empower wildcard search. In the inquiry stage, an information client questions a search term containing wildcard. For instance, Alice's specialist Bob may utilize the search query term "05/**/2018" to scan for all Alice's EHRs made amid the long stretch of May of 2018 and utilize the search term "*ache" to look for Alice's EHRs containing "cerebral pain", "headache" or "grief". But the system utilizes wildcard to denote the missing characters, and moreover, wildcard represent only one missing character of the keyword. To overcome this we introduce a method based on deterministic property of Cryptography algorithm which helps in fuzzy keyword search. To improve the security we also use Attribute based encryption for access control in user group scenario.

II. RELATED WORK

Searchable encryption in cloud computing has been area of interest for many researchers. There has been evolution from simple search by using keyword by Song, et al [1] to more advance level of searching to improve the user search experience. To improve query expressiveness many keyword search methods were introduced. Some are conjunctive keyword search where multiple words are collected and passed as a single query to server similar to the one presented by Goh et al[3] using DDH assumptions but the privacy of keywords were at stake. Similarly Ballard, et al [4], presented two schemes in the area of conjunctive keyword using symmetric key encryption. To check authenticity of the returned search results by the server verifiable search concept was introduced. Zheng, et al [8] worked on verifiable search using VABKS method, however, the verification process is costly. To improve the ease of search ranked search was introduced which reduced the communication traffic and increased the search efficiency. Cao, et al [10], introduced the rank search using MRSE scheme. Even though the relevant results were returned yet the results were based on keyword matches rather than the significance of different keywords. The traditional SE systems only worked on exact keyword match, but the incomplete or partial query keywords were not taken into consideration. To overcome this, fuzzy keyword search was introduced. This allows searching of query keywords with simple typos, and incomplete keywords. Li, et al[6] proposed a system based on string similarity and edit distance that allowed

small typos to be considered while searching. However, this required a predetermined set of keywords to be built which increased the storage complexity. Later on, Suga, et al[14] introduced wildcard search scheme based on bloom filter which represented the missing characters of keyword as '*', thus searching for the remaining characters of the keyword excluding the wildcard. However, in his method, the '*' denoted only single character.

In our proposed system, we have removed the necessity of representing the wildcard characters in place of missing characters. Moreover, the above method of wildcard search does not support dynamic update of files, where as our proposed system provides dynamic update to the files without the need of re encryption or without the need of generating new keys. Feature of presenting top k files is also added in our system to improve the search results.

III. PROPOSED SYSTEM

In this paper, we propose a fuzzy searchable encryption system to support incomplete or partial keyword queries which has several highly desirable features. We build our system using AES algorithm for encryption and Attribute based encryption to improve the access control privacy. The fuzzy search process depends greatly on deterministic property of AES algorithm.

Advantages of proposed work.

1) Privacy protection
2) Time based file accessing
3) Easy searching.
4) Difficult to crack
5) Highly secure.

Deterministic Property

Execution of encryption algorithm will always return the same encipher for the same input plaintext for the given key. Thus encrypting a character with same key will return the same ciphertext every time. We use this property in our system to conduct search on encrypted keywords as this allows flexible search on encrypted keywords while maintaining privacy.

To search over a plaintext we simply match the characters of the search term with the words present in the file. However, in case of encrypted file, the words are stored in encrypted form, so comparison of characters does not result in correct search result. To overcome this, we simply create a lookup table of encrypted keywords. These keywords are encrypted using deterministic encryption which returns the same encipher every time they are encrypted. We also encrypt the search token or query keyword with deterministic encryption. This allows comparison of encrypted keywords and query keywords. Hence if a valid existing query keyword is encrypted, then the resultant search token will be one of the existing keyword from the lookup table.

In this section, we propose a system that uses a secure and dynamic SSE algorithm for encryption which also helps in keyword character search. The proposed solution can use any dynamic Deterministic algorithm, so we opted for AES algorithm because of its simple and efficient characteristic along with Attribute Based Encryption (ABE) on the encryption and decryption module. The solution delegates the search functionality to CSP but with privacy in check. The AES algorithm is based on symmetric/ private key settings and so we can use only for single reader/writer settings or scenarios. The suggested architecture consists of six operational/functional entities: Keyword extractor, Key Authority, Encryption, Trapdoor generation, Search and Decryption. Functions of each block/entity are described as:

A. Key Authority

It generates and shares the keys for en/decryption to users and data owners. The key request is usually for encryption by data owner and decryption for data user. Two private keys are generated for encryption of documents and look up table. Separate private keys K1 & K2 are generated for both index table and document collection encryption. The keys are maintained on the user device as in symmetric settings same private key is used for en/decryption and also as it requires low computation overhead. We use SecureRandom Class of java.security package to generate keys. Strong and robust pseudo random number generator (PRNG) is used to provide implementation for any cryptography algorithms.

B. Key Extraction

This block or entity performs the necessary operations to pre-process a collection of files before the encryption. First the documents are allotted unique id. The id will start from 0 and increases serially. The documents are run through stop word filtering first so we can remove meaningless words from the document. In keyword extraction step, the document is searched for non-punctuation and non-blank characters ordered sequentially. A search of ASCII characters can also be easily tweaked. This results in uncompressed English text.

Usually to read the document characters, we have to first convert it to text file, but this increases the uploading process time, so we have used documents with .txt extension thus reducing the conversion time. The keyword extractor is implemented by pulling character from stream, checking if they are acceptable, and accumulating consecutive acceptable characters into a keyword. The count of appearance of words is maintained, and the words arriving most frequently are allotted as keywords for the document. This pre-processing step is required for creation of lookup table. Let F define a collection of files to be encrypted. The user has to list the keywords for documents from each collection F and build a lookup / index table listing the keywords and

the corresponding documents. In our system, we directly insert the encrypted keyword in the database table adjacent to their files instead of maintaining a separate lookup table.

C. Encryption

This entity guides the extracted keywords and their corresponding files through encryption. The lookup table along with the collection files is uploaded is input to this module. In our system, both are encrypted using AES algorithm and ABE technique. The user who is uploading the file is said to be the data owner. In our system, the data owner has a predefined attribute set during the registration process. The attribute is set and recorded for each user by ABE. The encryption module sends a key request to the key authority block for encryption. After receiving the key the system encrypts the files and lookup table. The block diagram for this module is given in Fig 3. Before encrypting the keywords, each character of the keyword is passed to hash functions to generate hash values. Each hashed value is then mapped to array data structure called bloom filter for performing search operations. Once the files are encrypted they are stored in the cloud server.

D. Trapdoor Generation

If a user wants to search files, he/she does by searching the keyword terms in the lookup table. To generate this encrypted search query keyword is called trapdoor generation. It is a token sent by the user to server to perform search operation on the enciphered lookup table. A user enters the query keyword. It is then passed through the hash function and later encrypted with the private key. This token is sent to server as a search request. The following steps are performed while trapdoor generation

Step 1: Query keyword is split by single character.
Step 2: The single character is then passed to hash function and corresponding hash value is calculated.
Step 3: Then each hash value is encrypted and appended with constant values to separate each character.
Step 4: Encipher is sent along with the attribute.
Step 5: This is sent to the server as a token.

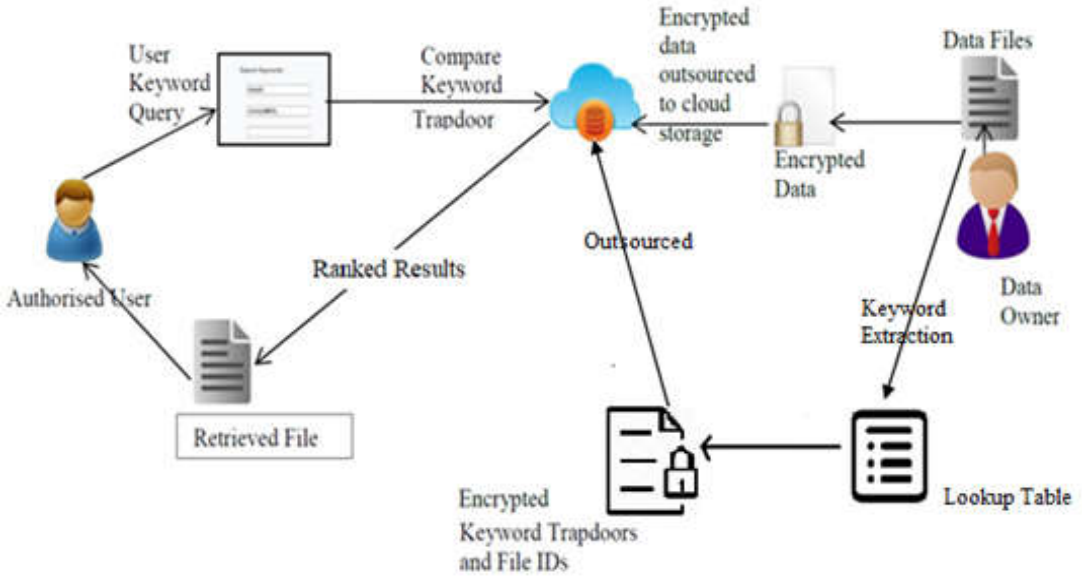

Fig. 1 Proposed System Architecture

E. Search Operation

The cloud server performs the search operation. This begins when the cloud server receives the search token from the user. The server then performs the comparison operation between the received token and the lookup table, if the encipher match then the corresponding document is fetched from the table and returned as a search result.

Thus incomplete or fuzzy query keyword can be used to search efficiently improving users search experience.

Step 1: Receive the token from the user.
Step 2: Compare the token with the lookup table keywords
Step 3: Compare the search token attribute with the keyword attribute in lookup table.
Step 4: If both are matched, then the files are retrieved.
Step 5: Sent the enciphered resultant files to the user.

F. Decryption

Once the search result is retrieved, the user downloads the retrieved result. The user requests the key for decryption to the key authority. Using the key the corresponding file is decrypted by the user.

IV. ALGORITHMS

Workflow of the proposed system

Taking the system goal into consideration, the developed system functionality can be achieved in 4 main task, they are:

1. Keyword Extractor Method
2. Encryption Algorithm
   - i)Document Encryption
   - ii)Keyword Encryption
3. Search Token Generation Scheme (as explained above)
4. Search Operation (as explained above).

1. Key Extraction Method

Step 1: Create a File ID for the documents to be uploaded on cloud.
Step 2: Create a stop words set.
Step 3: Read the file and remove the stop words from the files.
Begin loop
Step 4: For each word in the file count the frequency of occurrence.
Step 5: Update the counter for each word.
End loop
Step 6: Sort the order of the words based on the highest number of occurrence.
Step 7: Assign the top frequent words as keywords..

2. Encryption Algorithm

Step 1: Create file ID for the documents to be uploaded on cloud.
Step 2: Pass the document to the encryption modules
Step 3: Pass the generated keyword to encryption modules.

2.1) Document Encryption using AES Algorithm

Input: Plain Document
Output: Encrypted Document
The following are the algorithm steps:
Step 1: Derive round keys from cipher key.
Step 2: Represent the documents as state array of 16 bytes block.
begin loop
Step 3: Add initial round key to the starting state array.
Step 4: Perform the four operation of transformation in each round.
Step 5: Perform the tenth and final round of state manipulation
end loop
Step 6: Take out the final state array as enciphered data.

2.2) Keyword Encryption using AES Algorithm

The keyword generated for the file is passed through many steps before encryption.

1 .Keyword Splitting
2. Hash value generation
3. Deterministic Encryption

Step 1: The keywords generated from the keyword extractor are split with all possibilities.
Step 2: The split keywords are then represented as hash values using Hash map.
Step 3: The hash values for each divided keyword character set is generated.
Step 4: The split set of each keyword is then passed to AES encryption.
Step 5: Encipher of split keyword is then stored on the database table as lookup table.
Step 6: The attribute permission is also saved along with encipher in the lookup table

V. RESULTS

The following diagrams show the results of the search operation. When the user enters the incomplete query or if the characters of the keywords are missing, even then the search operation returns relevant files.
Consider there are keywords like ‘data’, ‘data-centric’, ‘database’, etc. stored in encrypted form in the lookup table. When the user enters the impartial query, the keyword is encrypted and sent as a token to the server and the server, takes the encipher of ‘dat’ and then compares with the keywords present in lookup table, if all the three characters match in the keyword, regardless of their position the encrypted files are sent. The Fig 2, shows the impartial query keyword ‘dat’ when entered retrieves all the files that have keywords with ‘dat’. The result of relevant files fetched is shown in Fig 3.

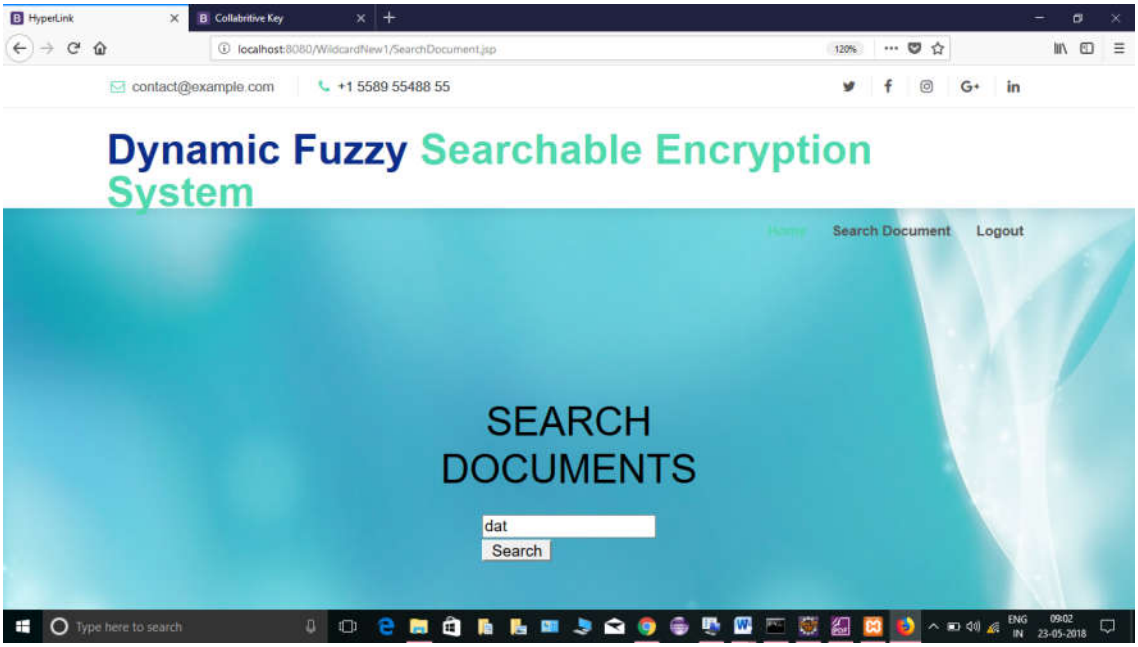

Fig. 2 Enter impartial query keyword

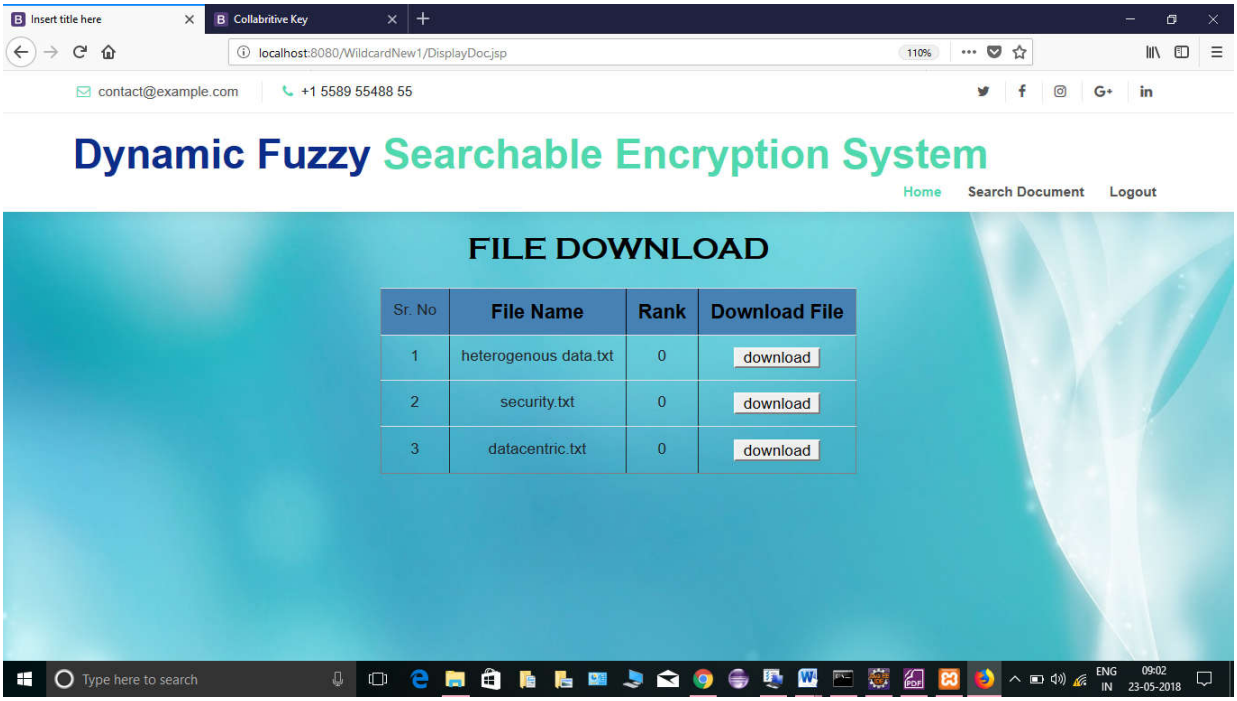

Fig. 3 Search result based on incomplete query keyword

VI. Conclusion

Cloud servers are untrusted third party organization, which might be semi-honest or curious about the data stored on it. To overcome the privacy issues, the data is stored in encrypted form on the server. However, storing the data in encrypted form hinders the search ability. To overcome this searchable encryption scheme using keyword is used. To overcome the problem of query expressiveness in our approach, we search on encrypted keywords without any wildcard representation. Our approach uses the deterministic property of AES for fuzzy keyword search. We use deterministic property of AES cryptography algorithm, which results in same ciphertext every time we use the same key on the plaintext. This property allows implementing fuzzy search on the encrypted keywords without the need of any wildcard character representation in the query keyword. Our scheme is simple yet efficient as the security of the keywords is maintained using the encrypted lookup table. Furthermore, along with search ability our scheme also supports dynamic update of file without the need of re encryption.

To provide privacy along with access control, we use Attribute based encryption for every user. The attribute of the user is encrypted along with the file for the access level determination. This provides privacy in user group scenarios providing data abstraction. Using rank algorithm, we sort the search results and present only the top ranked search files based on the user attribute. Thus removing the irrelevant files from the search result and improving search access time.

REFERENCES

[1] *Song D X, Wagner D, Perrig A, Practical techniques for searches on encrypted data, 2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, 2000, 44-55.*

[2] *Goh E J, Secure Indexes, IACR cryptology eprint archive, 2003, 216.*

[3] *Golle P, Staddon J, Waters B. Secure conjunctive keyword search over encrypted data. The 2nd International Conference on Applied Cryptography and Network Security, Yellow Mountain, China, 2004, 31-45.*

[4] *Ballard L, Kamara s, Monrose F. Achieving efficient conjunctive keyword searches over encrypted data. The 7th International Conference on Information and Communications Security, Beijing, China, 2005, 414-426.*

[5] *Faber S, Jarecki S, Krawczyk H, et al. Rich queries on encrypted data: beyond exact matches. The 20th European Symposium on Research in Computer Security, Vienna, Austria, 2015, 123-145.*

[6] *Li J, Wang Q, Wang C, et al. Fuzzy keyword search over encrypted data in cloud computing. The 29th IEEE International Conference on Computer, San Diego, USA, 2010, 441-445.*

[7] *Kuzu M, Islam M S, Kantarcioglu M. Efficient similarity search over encrypted data. The28th IEEE International Conference on Data Engineering, Washington, USA, 2012, 1156-1167.*

[8] *Zheng Q, Xu S, Ateniese G. VABKS: verifiable attribute-based keyword search over outsourced encrypted data. IEEE Conference on Computer Communications, Toronto, Canada, 2014, 522-530.*

[9] *Liu P, Wang J, Ma H, et al. Efficient verifiable public key encryption with keyword search based on KP-ABE. The 9th International Conference on Broadband and Wireless Computing Communication and Applications (BWCCA), Guangdong, China, 2014, 584-589.*

[10] *Cao N, Wang C,Li M, et al. Privacy preserving multi-keyword ranked search over encrypted cloud data. The 30th International Conference on Computer Communications, Shanghai, China, 2011, 829-837.*

[11] *Sun W, Wang B, Cao N, et al. Verifiable privacy-preserving multi-keyword text search in cloud supporting similarity based ranking. IEEE transactions on Parallel and Distributed Systems, 2014, Vol. 25, 3025-3035.*

[12] *Xia Z, Wang X, Sun X, et al. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE transctions on parallel and distributed systems, 2016, Vol. 27, 340-352.*

[13] *Chen C, Zhu X, Shen P, et al. An efficient privacy-preserving ranked keyword search method. IEEE Transactions on Parallel and Distributed Systems, 2016, Vol. 27, 951-963.*

[14] *Suga T, Nishide T, Sakurai K. Secure keyword search using Bloom filter with specified character position. International Conference on Provable Security. Springer Berlin Heidelberg, 2012, 235-252.*

[15] *Hu C, Han L. Efficient wildcard search over encrypted data. International Journal of Information Security. 2015, 1-9.*

[16] *Wang Y, Wang J, Chen X. Secure searchable encryption: a survey. Journal of Communications and Information Networks, 2016, Vol 1, 52-65.*

[17] *Kale P V, Welekar R. A survey on different techniques for encrypted cloud data. International Conference on Intelligent Computing and Control System, 2017, 245-247.*

[18] *Li B, Jiang J. Search on encrypted data: state of arts. Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), IEEE, 2016, 2022-2031.*

[19] *Zhang J, Semantic-based searchable encryption in cloud: issues and challenges. 2015 First International Conference on Computational Intelligence Theory, Systems and Applications, 2015, 163-165.*

[20] *Kamara S, Papamanthou C, Roeder T. Dynamic searchable symmetric encryption. Proceeding of ACM Conference on Computer and Communications Security(CCS), USA, 2012, 965-976.*

[21] *Bosch C, Hartel P, Jonker W, et al. A survey of provably secure searchable encryption. ACM Computing Surveys, Vol 47, 2014, 1-51.*