International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 05 | May -2017 www.irjet.net p-ISSN: 2395-0072

THE IMPLEMENTATION OF ALGORITHM TO PROVIDE MULTIKEYWORD RANK SEARCH OVER ENCRYPTED DATA

Mr. Ashwashil C. Lanjewar[1], Prof. Swapnil N. Sawalkar[2] , Prof. Harshal N.Datir[3]

[1]Student, Department of Information Technology, Sipna College of Engineering, Amravati, Maharashtra,
[2]Professor, Department of Information Technology, Sipna College of Engineering, Amravati, Maharashtra, India,
[3]Assistant Professor, Department of Information Technology, Sipna College of Engineering, Amravati, Maharashtra, India,

---

ABSTRACT - Due to the increasing popularity of cloud computing, more and more data owners are motivated to outsource their data to cloud servers for great convenience and reduced cost in data management. However, sensitive data should be encrypted before outsourcing for privacy requirements, which obsoletes data utilization like keyword-based document retrieval. In this paper, we present a secure multi-keyword ranked search scheme over encrypted cloud data, which simultaneously supports dynamic update operations like deletion and insertion of documents. We construct a special algorithm to provide efficient multi-keyword ranked search. The secure AES algorithm is utilized to encrypt the index and query vectors. The proposed scheme can achieve sub-linear search time and deal with the deletion and insertion of documents flexibly. Extensive experiments are conducted to demonstrate the efficiency of the proposed scheme.

Index Terms—Searchable encryption, multi-keyword ranked search, cloud computing.

1. INTORDUCTION

CLOUD computing has been considered as a new model of enterprise IT infrastructure, which can organize huge resource of computing, storage and applications, and enable users to enjoy ubiquitous, convenient and on-demand network access to a shared pool of configurable computing resources with great efficiency and minimal economic overhead [1]. Attracted by these appealing features, both individuals and enterprises are motivated to outsource their data to the cloud, instead of purchasing software and hardware to manage the data themselves.

Fig. no. 1.1: Cloud Computing Diagrammatical Representation

Despite of the various advantages of cloud services, outsourcing sensitive information (such as e-mails, personal health records, company finance data, government documents, etc.) to remote servers brings privacy concerns. The cloud service providers (CSPs) that keep the data for users may access users' sensitive information without authorization. A general approach to protect the data confidentiality is to encrypt the data before outsourcing [2]. However, this will cause a huge cost in terms of data usability. For example, the existing techniques on keyword-based information retrieval, which are widely used on the plaintext data, cannot be directly applied on the encrypted data. Downloading all the data from the cloud and decrypt locally is obviously impractical. In order to address the above problem, researchers have designed some general-purpose solutions with fully homomorphic encryption [3] or oblivious RAMs [4].

However, these methods are not practical due to their high computational overhead for both the cloud sever and user. On the contrary, more practical special purpose solutions, such as searchable encryption (SE) schemes have made specific contributions in terms of efficiency, functionality and security. Searchable encryption schemes enable the client to store the encrypted data to the cloud and execute keyword search over ciphertext domain.

So far, abundant works have been proposed under different threat models to achieve various search functionality, such as single keyword search, similarity search, multi-keyword boolean search, ranked search, multi-keyword ranked search, etc. Among them, multikeyword ranked search achieves more and more attention for its practical applicability.

Recently, some dynamic schemes have been proposed to support inserting and deleting operations on document collection. These are significant works as it is highly possible that the data owners need to update their data on the cloud server. But few of the dynamic schemes support efficient multikeyword ranked search. This paper proposes a secure tree-based search scheme over the encrypted cloud data, which supports multikeyword ranked search and dynamic operation on the document collection. Specifically, base64 and fuzzy approximation logic are used for encryption and query generation to provide multikeyword ranked search. Due to the, the proposed search scheme can flexibly achieve sub-linear search time and deal with the deletion and insertion of documents. In this paper, we focus on enabling effective yet privacy preserving fuzzy keyword search in Cloud Computing. To the best of our knowledge, we formalize for the first time the problem of effective fuzzy keyword search over encrypted cloud data while maintaining keyword privacy. Fuzzy keyword search greatly enhances system usability by returning the matching files when users' searching inputs exactly match the predefined keywords or the closest possible matching files based on keyword similarity semantics, when exact match fails. More specifically, we use edit distance to quantify keywords similarity and develop a novel technique, i.e., a wildcard-based technique, for the construction of fuzzy keyword sets. This technique eliminates the need for enumerating all the fuzzy keywords and the resulted size of the fuzzy keyword sets is significantly reduced. Based on the constructed fuzzy keyword sets, we propose an efficient fuzzy keyword search scheme. Through rigorous security analysis, we show that the proposed solution is secure and privacy-preserving, while correctly realizing the goal of fuzzy keyword search.

2. LITERATURE REVIEW

Searchable encryption schemes enable the clients to store the encrypted data to the cloud and execute keyword search over ciphertext domain. Due to different cryptography primitives, searchable encryption schemes can be constructed using public key based cryptography [5], [6] or symmetric key based cryptography [7], [8]. Song et al. [7] proposed the first symmetric searchable encryption (SSE) scheme, and the search time of their scheme is linear to the size of the data collection. Goh [8] proposed formal security definitions for SSE and designed a scheme based on Bloom filter. The search time of Goh's scheme is O (n), where n is the cardinality of the document collection. Curtmola et al. proposed two schemes (SSE-1 and SSE-2) which achieve the optimal search time. Their SSE-1 scheme is secure against chosen-keyword attacks (CKA1) and SSE-2 is secure against adaptive chosen-keyword attacks (CKA2). These early works are single keyword boolean search schemes, which are very simple in terms of functionality.

Afterward, abundant works have been proposed under different threat models to achieve various search functionality, such as single keyword search, similarity search, multi-keyword boolean search ranked search and multi-keyword ranked search etc. Multi-keyword boolean search allows the users to input multiple query keywords to request suitable documents. Among these works, conjunctive keyword search schemes only return the documents that contain all of the query keywords. Disjunctive keyword search schemes return all of the documents that contain a subset of the query keywords. Predicate search schemes are proposed to support both conjunctive and disjunctive search. All these multikeyword search schemes retrieve search results based on the existence of keywords, which cannot provide acceptable result ranking functionality. Ranked search can enable quick search of the most relevant data. Sending back only the top-k most relevant documents can effectively decrease network traffic.Some early works have realized the ranked search using order-preserving techniques, but they are designed only for single keyword search. Cao et al. realized the first privacy-preserving multi-keyword ranked search scheme, in which documents and queries are represented as vectors of dictionary size. With the "coordinate matching", the documents are ranked according to the number of matched query keywords.

3. PROPOSED WORK AND OBJECTIVES

A general approach to protect the data confidentiality is to encrypt the data before outsourcing. Searchable encryption schemes enable the client to store the encrypted data to the cloud and execute keyword search over ciphertext domain. So far, abundant works have been proposed under different threat models to achieve various search functionality, such as single keyword search, similarity search, multi-keyword boolean search, ranked search, multi-keyword ranked search, etc. Among them, multi-keyword ranked search achieves more and more attention for its practical applicability. Recently, some dynamic schemes have been proposed to support inserting and deleting operations on document collection. These are significant works as it is highly possible that the data owners need to update their data on the cloud server.

Huge cost in terms of data usability. For example, the existing techniques on keyword-based information retrieval, which are widely used on the plaintext data, cannot be directly applied on the encrypted data. Downloading all the data from the cloud and decrypt locally is obviously impractical. The fuzzy keyword search scheme returns the search results according to the following rules: 1) if the user's searching input exactly matches the pre-set keyword, the server is expected to return the files containing the keyword1; 2) if there exist typos and/or format inconsistencies in the searching

input, the server will return the closest possible results based on pre-specified similarity semantics.In computer science, approximate string matching (often colloquially referred to as fuzzy string searching) is the technique of finding strings that match a pattern approximately (rather than exactly).

The problem of approximate string matching is typically divided into two sub-problems: finding approximate substring matches inside a given string and finding dictionary strings that match the pattern approximately. One possible definition of the approximate string matching problem is the following: Given a pattern string and a text string, find a substring in T, which, of all substrings of T, has the smallest edit distance to the pattern P. A brute-force approach would be to compute the edit distance to P for all substrings of T, and then choose the substring with the minimum distance. However, this algorithm would have the running time $O(n^3 m)$.A better solution, which was proposed by Sellers[3], relies on dynamic programming. It uses an alternative formulation of the problem: for each position j in the text T and each position i in the pattern P, compute the minimum edit distance between the i first characters of the pattern, , and any substring of T that ends at position j. For each position j in the text T, and each position i in the pattern P, go through all substrings of T ending at position j, and determine which one of them has the minimal edit distance to the i first characters of the pattern P. Write this minimal distance as E(i, j). After computing E (i, j) for all i and j, we can easily find a solution to the original problem: it is the substring for which E (m, j) is minimal (m being the length of the pattern P.) Computing E (m, j) is very similar to computing the edit distance between two strings. In fact, we can use the Levenshtein distance computing algorithm for E(m, j), the only difference being that we must initialize the first row with zeros, and save the path of computation, that is, whether we used E(i − 1,j), E(i,j − 1) or E(i − 1,j − 1) in computing E(i, j). In the array containing the E(x, y) values, we then choose the minimal value in the last row, let it be $E(x_2, y_2)$, and follow the path of computation backwards, back to the row number 0. If the field we arrived at was $E(0, y_1)$, then $T[y_1 + 1] ... T[y_2]$ is a substring of T with the minimal edit distance to the pattern P. Computing the E(x, y) array takes O(mn) time with the dynamic programming algorithm, while the backwards-working phase takes O(n + m) time. Here we are using AES for Encryption as AES AES is based on a design principle known as a substitution-permutation network, a combination of both substitution and permutation, and is fast in both software and hardware. Unlike its predecessor DES, AES does not use a Feistel network. AES is a variant of Rijndael which has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. By contrast, the Rijndael specification per se is specified with block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits.

Notation :- A_k = Admin key.
K = Keywords entered in the search query.

Proposed Algorithm

i. Keyword entered in the search query **do**
ii. **If** A_k = = K then **do**
iii. AES encryption on K and send it to base64
iv. **If** base64 returns the base64 string **then** Load all the files detail of the database to the server in the hash list and call the fuzzy search() function
v. **If** the Fuzzy search () function returns top-k result **then** send the encrypted files to the user.

If the user is authorized by the data owner then by using this algorithm user will be able to search over encrypted data and can access the encrypted file.

4. IMPLEMENTATION

MODULES

- Data Owner Module
- Data User Module
- Cloud server and Encryption Module
- Rank Search Module

MODULES DESCRIPTION

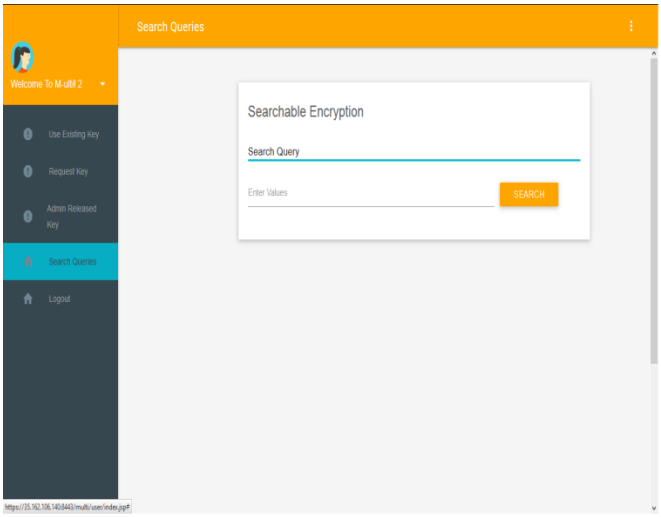
4.1. Data Owner Module

This module helps the owner to register those details and also include login details. This module helps the owner to upload his file with encryption using AES algorithm. This ensures the files to be protected from unauthorized user. Data owner has a collection of documents F that he wants to outsource to the cloud server in encrypted form while still keeping the capability to search on them for effective utilization. In our scheme, the data owner firstly builds a secure searchable

tree index I from document collection F, and then generates an encrypted document collection C for F. Afterwards, the data owner outsources the encrypted collection C and the secure index I to the cloud server, and securely distributes the key information of and document decryption to the authorized data users. Besides, the data owner is responsible for the update operation of his documents stored in the cloud server. While updating, the data owner generates the update information locally and sends it to the server.

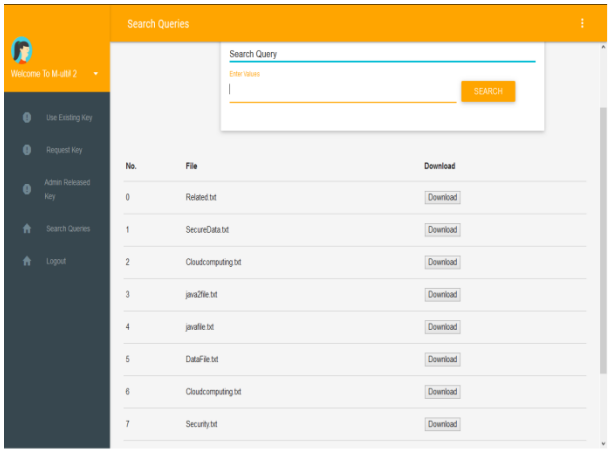
4.2. Data User Module

This module includes the user registration login details. This module is used to help the client to search the file using the multiple keywords concept and get the accurate result list based on the user query.



Screenshot no. 4.2.4: Query Search Page

This is the section for the user where the user will be able to search over the encrypted cloud data, by entering the keywords in the search query. The keywords entered will be searched over the encrypted cloud data by using searchable encryption.



Screenshot no. 4.2.5: Page of Searched Query Result

This is the interface for the user where users will get the result for the search query which user has searched by entering the keyword in the search query.

4.3. Cloud Server and Encryption Module

This module is used to help the server to encrypt the document using AES Algorithm and to convert the encrypted document to the file and then activation code send to the user for download. Cloud server stores the encrypted document for data owner. Upon receiving the Admin key from the data user, the cloud server executes search over the hash list, and finally returns the corresponding collection of top- k ranked encrypted documents. Besides, upon receiving the update

information from the data owner, the server needs to update the index I and document collection C according to the received information. The cloud server in the proposed scheme is considered as "honest-but-curious", which is employed by lots of works on secure cloud data search

4.4. Rank Search Module

These modules ensure the user to search the files that are searched frequently using rank search. This module allows the user to download the file using his secret key to decrypt the downloaded data. This module allows the Owner to view the uploaded files and downloaded files. The proposed scheme is designed to provide not only multi-keyword query and accurate result ranking, but also dynamic update on document collections. The scheme is designed to prevent the cloud server from learning additional information about the document collection and the query.

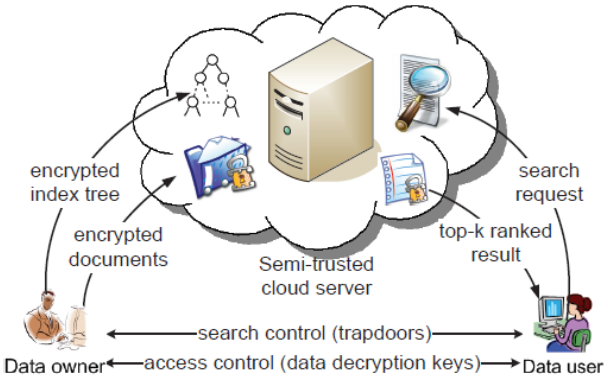


Fig. 4.4.1. The architecture of ranked search over encrypted cloud data

4.5. Design Goals

To enable secure, efficient, accurate and dynamic multikeyword ranked search over outsourced encrypted cloud data under the above models, our system has the following design goals.
Dynamic: The proposed scheme is designed to provide not only multi-keyword query and accurate result ranking, but also dynamic update on document collections.

Search Efficiency: The scheme aims to achieve sublinear search efficiency by exploring a special tree-based index and an efficient search algorithm.

Privacy-preserving: The scheme is designed to prevent the cloud server from learning additional information about the document collection, the index tree, and the query.

5. ADVANTAGES

- Multi-keyword ranked search over encrypted cloud data.
- Ranked based search.
- Price Pay for only the resources used.
- Security Cloud instances are isolated in the network from other instances for improved security.
- Performance Instances can be added instantly for improved performance. Clients have access to the total resources of the Cloud's core hardware.
- Scalability Auto-deploy cloud instances when needed.
- Uptime uses multiple servers for maximum redundancies. In case of server failure, instances can be automatically created on another server.
- Control Able to login from any location. Server snapshot and a software library lets you deploy custom instances.
- Traffic Deals with spike in traffic with quick deployment of additional instances to handle the load.

CONCLUSION

In this paper, a secure, efficient and dynamic search scheme is proposed, which supports not only the accurate multi-keyword ranked search but also the dynamic deletion and insertion of documents. In addition, the parallel search process can be carried out *to* further reduce the time cost. The security of the scheme is protected against threat by using the secure algorithm. Experimental results demonstrate the efficiency of our proposed scheme. There are still many challenge problems in symmetric SE schemes. In the proposed scheme, the data owner is responsible for generating

updating information and sending them to the cloud server. It could be a meaningful but difficult future work to design a dynamic searchable encryption scheme whose updating operation can be completed by cloud server only, meanwhile reserving the ability to support multi-keyword ranked search. In addition, as the most of works about searchable encryption, our scheme mainly considers the challenge from the cloud server. Actually, there are many secure challenges in a multi-user scheme. In the future works, we will try to improve the SE scheme to handle these challenge problems.

REFERENCE

[1] K. Ren, C.Wang, Q.Wang et al., "Security challenges for the public cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69–73, 2012.

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

[3] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[4] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," Journal of the ACM (JACM), vol. 43, no. 3, pp. 431–473, 1996.

[5] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in Advances in Cryptology - Eurocrypt 2004. Springer, 2004, pp. 506–522.

[6] D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. E. Skeith III, "Public key encryption that allows pir queries," in Advances in Cryptology-CRYPTO 2007. Springer, 2007, pp. 50–67.

[7] 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. IEEE, 2000, pp. 44 – 55.

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

[9] 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 Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security. ACM, 2013, pp. 71–82.

[10] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012, pp. 965–976.

[11] S. Kamara and C. Papamanthou, "Parallel and dynamic searchable symmetric encryption," in Financial Cryptography and Data Security. Springer, 2013, pp. 258–274.