TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.3, March 2014, pp. 2104 ~ 2109
DOI: http://dx.doi.org/10.11591/telkomnika.v12i3.4357

Conjunctive Fuzzy Keyword Search Over Encrypted Data in Cloud Computing

Jianhua Yu[1], Jin Li[2], Xueli Wang[1], Wei Gao[3]
[1]School of Mathematical Sciences, South China Normal University, Guangzhou, P.R.China
[2]School of Computer Science and Education Software, Guangzhou University, Guangzhou, P.R.China
[3]School of Mathematics and Statistics, Ludong University, Yantai, P.R.China

Abstract

As Cloud Computing becomes prevalent, more and more sensitive information are centralized into the cloud. For the protect data privacy, sensitive data usually have to be encrypted before outsourcing, which makes efficient data utilization a very challenging task. Although traditional searchable encryption schemes allow a user to search over encrypted data through keywords and selectively retrieve files of interest, these techniques only support exact keyword search. There is no tolerance of minor typos or format inconsistencies which, on the other hand, are typical user searching behavior and happen very frequently. This significant drawback makes existing techniques unsuitable in Cloud Computing as it greatly affects system usability, which results in frustrating searching experience and low system efficiency. In this paper, for the first time we formalize and solve the problem of efficient conjunctive fuzzy keyword search over encrypted cloud data while maintaining keyword privacy. Conjunctive 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. In our solution, we exploit edit distance to quantify keywords similarity. Through rigorous security analysis, we show that our proposed solution is privacy-preserving.

Keywords*: certificateless cryptography, certificateless threshold decryption, provable security, random oracle model, bilinear pairing*

1. Introduction

As Cloud computing is an increasingly popular and enticing computing model which enables users to outsource their data to the cloud servers. By storing their data into the cloud, the data owners can be relieved from the burden of data storage and maintenance so as to enjoy the on-demand high quality data storage service. Due to its flexible elasticity, efficiency, and low costs, an increasing number of web services are gradually abandon the traditional in-house datacenter and opting for the cloud computing.

Promising as it is, cloud computing also brings forth new challenges for data security and privacy when data owner outsources sensitive data on cloud servers. It follows that sensitive data usually should be encrypted prior to outsourcing for data privacy and combating unsolicited accesses. However, data encryption makes efficient data utilization a very challenging task given that there could be a large amount of outsourced data files. Usually, the individual users want to only retrieve certain specific data files they are interested in during a given session. Therefore, how to efficiently retrieve useful information from the large encrypted database becomes an important problem in cloud computing.

One of the most popular solutions is to selectively retrieve files through keyword-based search instead of retrieving all the encrypted files back which is completely impractical in cloud computing scenarios. Such keyword-based search technique allows users to selectively retrieve files of interest and has been widely applied in plaintext search scenarios, such as Google search [1]. Unfortunately, data encryption restricts server's ability to perform keyword search and thus makes the traditional plaintext search methods unsuitable for Cloud Computing. Besides, data encryption also demands the protection of keyword privacy since keywords usually contain important information related to the data files. Although encryption of keywords can protect keyword privacy, it further renders the traditional plaintext search techniques useless in this scenario.

Received September 1, 2013; Revised September 17, 2013; Accepted October 18, 2013

To securely search over encrypted data, searchable encryption techniques have been developed in recent years [2-10]. Searchable encryption schemes usually build up an index for each keyword of interest and associate the index with the files that contain the keyword. By integrating the trapdoors of keywords within the index information, effective keyword search can be realized while both file content and keyword privacy are well-preserved. Although allowing performing searches securely and effectively, most the existing searchable encryption techniques do not suit for cloud computing scenario since they support only exact keyword search. Whereas it is quite common that users' searching input might not exactly match those pre-set keywords due to the possible typos, such as Illinois and Ilinois, representation inconsistencies, such as PO BOX and P.O. Box, and/or her lack of exact knowledge about the data. In [11], Li et al. firstly formalize the problem of fuzzy keyword search over encrypted data in cloud computing. However, the scheme in [11] only supports single fuzzy keyword search. If the users want to retrieve the files that contain a set of keywords, he has to repeatedly implement the protocol for several times, which is rather inefficient.

In this paper, we focus on enabling effective yet privacy- preserving conjunctive fuzzy keyword search in Cloud Computing. To the best of our knowledge, we formalize for the first time the problem of effective conjunctive fuzzy keyword search over encrypted cloud data while maintaining keyword privacy. Conjunctive 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, for the construction of fuzzy keyword sets. Based on the constructed fuzzy keyword sets, we propose an efficient conjunctive 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 conjunctive fuzzy keyword search.

The rest of paper is organized as follows: Section 2 summarizes the features of related work. Section 3 introduces the system model, threat model, and the design goal and briefly describes some necessary for the techniques. Section 3 shows a straightforward construction of fuzzy keyword search scheme. Section 4 provides the detailed description of our proposed schemes, including the efficient constructions of fuzzy keyword set and fuzzy keyword search scheme. Section 5 presents the security analysis. Finally, Section 6 concludes the paper.

2. Related Work

The notion of Searches on Encrypted Data [3] was first proposed by Song et al. It deals with search problems between a user and an untrusted server, also they provide some practical solutions. Public Key Encryption with Keyword Search (PEKS) [5] was first proposed by Boneh and Boyen in 2004. Since then, keywords search problems have been divided into two parts. One is private key model, and the other is public key model. Traditional searchable encryption [2-8], [10] has been widely studied in cryptography. Among those works, most are focused on efficiency improvements and security definition formalizations. Goh [4] proposed to use Bloom filters to construct the indexes for the data files. To achieve more efficient search, Chang et al. [7] and Curtmola et al. [8] both proposed similar "index" approaches, where a single encrypted hash table index is built for the entire file collection. In the index table, each entry consists of the trapdoor of a keyword and an encrypted set of file identifiers whose corresponding data files contain the keyword. As a complementary approach, Boneh et al. [5] presented a public-key based searchable encryption scheme, with an analogous scenario to that of [3]. Note that all these existing schemes support only exact keyword search, and thus are not suitable for Cloud Computing.

The importance of fuzzy search has received attention in the context of plaintext searching in information retrieval community [12-14]. They addressed this problem in the traditional information access paradigm by allowing user to search without using try-and-see approach for finding relevant information based on approximate string matching. At the first glance, it seems possible for one to directly apply these string matching algorithms to the context of searchable encryption by computing the trapdoors on a character base within an alphabet. However, this trivial construction suffers from the dictionary and statistics attacks and fails to achieve the search privacy. In [11], Li et al. firstly formalize the problem of fuzzy keyword search over encrypted data in cloud computing. However, the scheme in [11] only supports

single fuzzy keyword search. If the users want to retrieve the files that contain a set of keywords, he has to repeatedly implement the protocol for several times, which is rather inefficient.

Private matching [15], as another related notion, has been studied mostly in the context of secure multiparty computation to let different parties compute some function of their own data collaboratively without revealing their data to the others. These functions could be intersection or approximate private matching of two sets. The private information retrieval [16] is an usual technique to retrieve the matching items secretly, which has been widely applied in information retrieval from database and usually incurs unexpectedly computation complexity.

More recently, Park, Kim and Lee proposed the first public-key searchable encryption schemes [5, 6, 17] that allow secure conjunctive keyword searches [18]. Their constructions, based on the Bilinear Decision Diffie-Hellman (BDDH) and Bilinear Decision Diffie-Hellman Inversion (BDDHI) assumptions, have constant sized trapdoors and are more efficient than the schemes presented in [19].

3. PreliminarieS

3.1. Edit Distance

There are several methods to quantitatively measure the string similarity. In this paper, we resort to the well-studied edit distance [19] for our purpose. The edit distance $ed(w, w')$ between two words w and w' is the number of operations required to transform one of them into the other. The three primitive operations are 1) Substitution: changing one character to another in a word; 2) Deletion: deleting one character from a word; 3) Insertion: inserting a single character into a word. Given a keyword w, we let $S_{s,d}$ denote the set of words w' satisfying $ed(w, w') \leq d$ for a certain integer d.

3.2. Bloom Filters

A Bloom filter [21] represents a set of $S = \{s_1, \ldots, s_n\}$ of n elements and is represented by an array of m bits. All array bits are initially set to 0. The filter uses r independent hash functions $H_1, \ldots, H_n$, where $H_i: \{0,1\} \rightarrow [1, m]$ for $i \in [1, r]$. For each element $s \in S$, the array bits at positions $H_1(s), \ldots, H_r(s)$ are set to 1. A location can be set to 1 multiple times, but only the first is noted. To determine if an element x belongs to the set S, we check the bits at positions $H_1(x), \ldots, H_r(x)$. If all the checked bits are 1's, then x is considered a member of the set. There is, however, some probability of a false positive, in which x appears to be in S but actually is not. False positives occur because each location may have also been set by some element other than s. On the other hand, if any checked bits are 0, then x is definitely not a member of S.

3.3. Model

We assume the standard model for symmetric searchable encryption schemes (as used in [3-4], [22-23]), which includes a client and a server that can be trusted to interact in a protocol, but not to abstain from attempting to learn information that is notexplicitly released by the client. The client has a set of m documents $D = \{D_1, \ldots, D_m\}$ that she wishes to store on the server in encrypted form, while still retaining the ability to search through them. To do so, she first generates an index I_i for each document D_i and stores both the index and the encrypted document $E(D_i)$ on the server. Here, we assume that E is an arbitrary symmetric encryption scheme, such as AES [24], and is independent of our index constructions. To search the document collection for a given keyword w, the client generates and sends a trapdoor T to the server who proceeds to search each index for w. The server then returns the appropriate set of documents to the client.

In this work, we also assume the standard model for conjunctive fuzzy keyword searches over encrypted data. Using edit distance, the definition of conjunctive fuzzy keyword search can be formulated as follows: Given a collection of m encrypted data files $D = \{D_1, \ldots, D_m\}$ stored in the cloud server, we associate a list of keywords $W = \{S_{w_{i1}}, \ldots, S_{w_{il}}\}$ with each document D_i($S_{w,d}$ denote the set of words w' satisfying $ed(w, w') \leq d$ for a certain integer d) and a searching input $W = (w_1, \ldots, w_d)$, the execution of conjunctive fuzzy keyword search returns a set of file IDs whose corresponding data files possibly contain the word $W = (w_1, \ldots, w_d)$.

3.4. Secure Conjunctive Fuzzy Keyword Search (SCFKS)

Let $W = (w_1, \ldots, w_d)$ be a collection of keyword lists. A SCFKS scheme consists of four probabilistic polynomial-time algorithms:

a) Keygen(1^k): is a probabilistic key generation algorithm that is executed by the client in order to instantiate the scheme. It takes as input a security parameter k, and returns a secret key K.

b) Trapdoor(K, W): is executed by the client to generate a trapdoor for a given conjunction of keywords. It returns a trapdoor T_W for the conjunction.

$$W = (w_1 \wedge \ldots \wedge w_d).$$

c) Build Index (K, D): is executed by the client to construct an index. It takes as input a secret key K and a document D. It returns D's index I_D.

d) Search Index(I_D, T_w): is executed by the server on behalf of the client to search for the occurrence of a conjunction in an index. It takes as input an index $I_D = BuildIndex(K, D)$, where a trapdoor T_w for a conjunction $W = (w_1 \wedge \ldots \wedge w_d)$.

4. Construction

We use a Bloom filter as a per document index to track words in each document [4]. Our construction makes use of r pseudo-random functions $H_i: \{0,1\}^s \times \{0,1\}^*, i = 1, \ldots, r$, we write $H_i^k(w = H_i(k \| w)$. We now describe our index.

Keygen (1^k): Given a security parameter s, generate a secret key $K = (k_1, \ldots, k_r) \leftarrow R\{0,1\}^{st}$ (We write $x \leftarrow RX$ to represent a random variable x being drawn uniformly from a set X).

Trapdoor (K, W): Given the master key $K = (k_1, \ldots, k_r) \in \{0,1\}^{st}$ and $W = (w_1, \ldots, w_d)$, output the trapdoor for word W as $T_W = (T_{w_1}, \ldots, T_{w_d})$, where $T_{w_i} = (H_1^{k_1}(w_i), \ldots, H_r^{k_r}(w_i)) \in \{0,1\}^{sr}, i \in [1, d]$.

Build Index (K, D): The input is the document D comprising of an unique identifier (name) $D_{id} \in \{0,1\}^n$ and a list of words $\{w_1, \ldots, w_p\} \in \{0,1\}^{np}$ and $K = (k_1, \ldots, k_r) \in \{0,1\}^{sr}$.

a) For each unique word w_i for $i \in [1, p]$, compute the fuzzy set:

We let $S_{w_i,d}$ denote the set of words w_i' satisfying $ed(w_i, w_i') \leq d$ for a certain integer d,

b) the trapdoor: for $w_i' \in S_{w_i,d}$ in D_{id}, compute:

$$(x_1 = H_1^{k_1}(w_i'), \ldots, x_r = H_r^{k_r}(w_i')) \in \{0,1\}^{sr},$$

(c) the codeword: $(y_1 = H_1^{x_{11}}(D_{id}), \ldots, y_r = H_r^{x_{r1}}(D_{id})) \in \{0,1\}^{sr}$,

(d) insert the codeword $y_1, \ldots, y_r$ into document D_{id}'s Bloom filter BF.

Output $I_{D_{id}} = (D_{id}, BF)$ as the index for D_{id}.

SearchIndex(T_W, I_D): The input is the trapdoor $T_W = (T_{w_1}, \ldots, T_{w_d})$ for $W = (w_1, \ldots, w_d)$ and the index $I_{D_{id}} = (D_{id}, BF)$ for document D_{id}.

a) Compute the codeword for w_i in D_{id}:

$$(y_1 = H_1^{x_{11}}(D_{id}), \ldots, y_r = H_r^{x_{r1}}(D_{id})) \in \{0,1\}^{sr}, i \in [1, d].$$

b) Test if BF contains 1's in all r locations denoted by $y_1, \ldots, y_r$.

c) If so, output 1; Otherwise, output 0.

5. Security

In this section, we analyze security of the proposed conjunctive fuzzy keyword search scheme Theorem 1: The fuzzy keyword search scheme is secure regarding the search privacy. Proof: In the wildcard-based scheme, the computation of index and request of the same keyword is identical. Therefore, we only need to prove the index privacy by using reduction. Suppose the searchable encryption scheme fails to achieve the index privacy against the

indistinguishability under the chosen keyword attack, which means there exists an algorithm $\mathcal{A}$ who can get the underlying information of keyword from the index. Then, we build an algorithm A' that utilizes $\mathcal{A}$ to determine whether some function $f'(\cdot)$ is a pseudo-random function such that $f'(\cdot)$ is equal to $f(sk,\cdot)$ or a random function. A' has an access to an oracle $O_{f'(\cdot)}$ that takes as input secret value x and returns $f(x)$. Upon receiving any request of the index computation, A' answers it with request to the oracle $O_{f'(\cdot)}$. After making these trapdoor queries, the adversary outputs two sets of challenge keywords $W_0 = (w_{01}, \ldots, w_{0d})$ and $W_1 = (w_{11}, \ldots, w_{1d})$ with the same length and edit distance, which can be relaxed by adding some redundant trapdoors. A' picks one random $b \in \{0,1\}$ and sends W_b to the challenger. Then, A' is given a challenge value y, which is either computed from a pseudo-random function $f(sk,\cdot)$ or a random function. A' sends y back to $\mathcal{A}$, who answers with $b \in \{0,1\}$. Suppose $\mathcal{A}$ guesses b correctly with nonnegligible probability, which indicates that the value is not randomly computed. Then, A makes a decision that $f'(\cdot)$ is a pseudo-random function. As a result, based on the assumption of the indistinguishability of the pseudo-random function from some real random function, $\mathcal{A}$ at most guesses b correctly with approximate probability $\frac{1}{2}$. Thus, the search privacy is obtained.

6. Conclusion

In this paper, for the first time we formalize and solve the problem of supporting efficient yet privacy-preserving conjunctive fuzzy search for achieving utilization of remotely stored encrypted data in Cloud Computing. Through rigorous security analysis, we show that our proposed solution is secure and privacy-preserving.

Acknowledgements

This work was supported by the National Nature Science Foundation of China (No.61202475, 61100224 and 11171222).

References

[1] Google. Britney spears spelling correction. http://www.google.com/jobs/ britney.html. 2009.
[2] Bellare M, Boldyreva A, O'Neill A. *Deterministic and efficiently searchable encryption.* Proceedings of Crypto 2007; 4622 of LNCS. Springer-Verlag.
[3] Song D, Wagner D, Perrig A. *Practical techniques for searches on encrypted data*. Proc. of IEEE Symposium on Security and Privacy. 2000.
[4] Goh EJ. Secure indexes, Cryptology ePrint Archive. http://eprint.iacr. org/. 2003/216.
[5] Boneh D, Crescenzo GD, Ostrovsky R, Persiano G. *Public key encryption with keyword search.* Proc. of EUROCRYP. 2004.
[6] Waters B, Balfanz D, Durfee G, Smetters D. *Building an encrypted and searchable audit log.* Proc. of 11th Annual Network and Distributed System. 2004.
[7] Chang YC, Mitzenmacher M. *Privacy preserving keyword searches on remote encrypted data*. Proc. of ACNS. 2005.
[8] Curtmola R, Garay JA, Kamara S, Ostrovsky R. *Searchable symmetric encryption: improved definitions and efficient constructions*. Proc. of ACM CCS. 2006.
[9] Boneh D, Waters B. *Conjunctive, subset, and range queries on encrypted data*. Proc. of TCC. 2007: 535-554.
[10] Bao F, Deng R, Ding X, Yang Y. *Private query on encrypted data in multi-user settings*. Proc. of ISPEC. 2008.
[11] Li J, Wang Q, Wang C, Cao N, Ren K, Lou W. *Fuzzy keyword search over encrypted data in cloud computing*. Proc. of IEEE INFOCOM. 2010.
[12] Li C, Lu J, Lu Y. *Efficient merging and filtering algorithms for approximate string searches*. Proc. of ICDE. 2008.
[13] Behm A, Ji S, Li C, Lu J. *Space-constrained gram-based indexing for efficient approximate string search.* Proc. of ICDE. 2009.
[14] Ji S, Li G, Li C, Feng J. *Efficient interactive fuzzy keyword search*. Proc. of WWW. 2009.
[15] Feigenbaum J, Ishai Y, Malkin T, Nissim K, Strauss M, Wright RN. *Secure multiparty computation of approximations*. Proc. ofICALP. 2001.
[16] Ostrovsky R. Software protection and simulations on oblivious rams, Ph.D dissertation, Massachusetts Institute of Technology. 1992.

[17] Davis D, Monrose F, Reiter M. *Time-scoped searching of encrypted audit logs*. 6th International Conference on Information and Communications Security (ICICS) of Lecture Notes in Computer Science. Springer-Verlag. 2004; 3269: 532-545.
[18] Park D, Kim K, Lee P. *Public key encryption with conjunctive field keyword search*. 5th International Workshop on Information Security Applications (WISA), of Lecture Notes in Computer Science, Springer-Verlag. 2004; 3325: 73-86.
[19] Jansen B, Spink A, Bateman J, Saracevic T. Real life information retrieval: a study of user queries on the web. SIGIR Forum. 1998; 32(1): 5-17.
[20] Levenshtein V. *Binary codes capable of correcting spurious insertions and deletions of ones*. Problems of Information Transmission. 1965; 1(1): 8-17.
[21] Bloom B. *Space/time trade-offs in hash coding with allowable errors*. Communications of the ACM. 1970; 13(7): 422-426.
[22] Chang Y, Mitzenmacher M. *Privacy preserving keyword searches on remote encrypted data.* Third International Conference on Applied Cryptography and Network Security (ACNS), of Lecture Notes in Computer Science, Springer-Verlag. 2005; 3531: 442-455.
[23] Golle P, Staddon J, Waters B. Secure conjunctive keyword search over encrypted data, in: Applied Cryptography and Network Security Conference (ACNS), of Lecture Notes in Computer Science, Springer-Verlag. 2004: 3089: 31-45.
[24] Federal Information Processing Standards. Advanced Encryption Standard (AES) -FIPS 197. 2001.