Fuzzy Keyword Search over Encrypted Data in Cloud Computing

[1]Seemakurthi Durga Prasad, [2]P. Venkata Narayana, [3]J. Karthik
[1,2]Dept. of IT & Engg, St. Ann's College of Engineering & Technology, Affiliated to JNTU Kakinada

Abstract
As Cloud Computing becomes prevalent, more and more sensitive information are being centralized into the cloud. Although traditional searchable encryption schemes allow a user to securely search over encrypted data through keywords and selectively retrieve files of interest, these techniques support only exact keyword search. In this paper, for the first time we formalize and solve 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. In our solution, we exploit edit distance to quantify keywords similarity and develop two advanced techniques on constructing fuzzy keyword sets, which achieve optimized storage and representation overheads. We further propose a brand new symbol-based tire-traverse searching scheme, where a multi-way tree structure is built up using symbols transformed from the resulted fuzzy keyword sets. Through rigorous security analysis, we show that our proposed solution is secure and privacy-preserving, while correctly realizing the goal of fuzzy keyword search. Extensive experimental results demonstrate the efficiency of the proposed solution.

Keywords
String Match Algorithm, System Architecture

I. Introduction

As Cloud Computing becomes prevalent, more and more sensitive information are being centralized into the cloud, such as emails, personal health records, government documents, etc. 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. However, the fact that data owners and cloud server are not in the same trusted domain may put the outsourced data at risk, as the cloud server may no longer be fully trusted. It follows that sensitive data usually should be encrypted prior to outsourcing for data privacy and combating unsolicited accesses. However, data encryption makes effective data utilization a very challenging task given that there could be a large amount of outsourced data files. Moreover, in Cloud Computing, data owners may share their outsourced data with a large number of users. The individual users might want to only retrieve certain specific data files they are interested in during a given session. One of the most popular ways 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. Unfortunately, data encryption restricts user's ability to perform keyword search and thus makes the traditional plaintext search methods unsuitable for Cloud Computing. Besides this, 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.

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.

II. Problem Formulations

A. System Model

In this paper, we consider a cloud data system consisting of data owner, data user and cloud server. Given a collection of n encrypted data files $C = (F_1, F_2 \ldots F_N)$ stored in the cloud server, a predefined set of distinct keywords $W = \{w_1, w_2, \ldots, w_p\}$, the cloud server provides the search service for the authorized users over the encrypted data C. We assume the authorization between the data owner and users is appropriately done. An authorized user types in a request to selectively retrieve data files of his/her interest. The cloud server is responsible for mapping the searching request to a set of data files, where each file is indexed by a file ID and linked to a set of keywords. 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 (to be formally defined in section III-D). An architecture of fuzzy.

B. Threat Model

We consider a semi-trusted server. Even though data files are encrypted, the cloud server may try to derive other sensitive information from users' search requests while performing keyword-based search over C. Thus, the search should be conducted in a secure manner that allows data files to be securely retrieved while revealing as little information as possible to the cloud server. In this paper, when designing fuzzy keyword search scheme, we will follow the security definition deployed in the traditional searchable encryption [8]. More specifically, it is required that nothing should

be leaked from the remotely stored files and index beyond the outcome and the pattern of search queries.

C. Design Goals

In this paper, we address the problem of supporting efficient yet privacy-preserving fuzzy keyword search services over encrypted cloud data. Specifically, we have the following goals:

1. to explore new mechanism for constructing storage efficient fuzzy keyword sets
2. to design efficient and effective fuzzy search scheme based on the constructed fuzzy keyword sets
3. to validate the security of the proposed scheme.

D. Preliminaries

Edit Distance There are several methods to quantitatively measure the string similarity. In this paper, we resort to the Well-studied edit distance [16] for our purpose. The edit distance ed(w1, w2) between two words w1 and w2 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 Sw,d denote the set of words w_ satisfying ed(w,w_) ≤ d for a certain integer d.

III. Related Work

Fuzzy Keyword Search Using edit distance, the definition of fuzzy keyword search can be formulated as follows: Given a collection of n encrypted data files $C = (F_1, F_2, \ldots, F_N)$ stored in the cloud server, a set of distinct keywords $W = \{w_1, w_2, \ldots, w_p\}$ with predefined edit distance d, and a searching input (w, k) with edit distance k ($k \leq d$), the execution of fuzzy keyword search returns a set of file IDs whose corresponding data files possibly contain the word w, denoted as FIDw: if w = wi ∈ W, then return FIDwi ; otherwise, if w _∈ W, then return {FIDwi }, where ed(w,wi) ≤ k. Note that the above definition is based on the assumption that $k \leq d$. In fact, d can be different for distinct Keywords and the system will return {FIDwi} satisfying ed(w,wi) ≤ min{k, d} if exact match fails.

IV. The Straightforward Approach

Before introducing our construction of fuzzy keyword sets, we first propose a straightforward approach that achieves all the functions of fuzzy keyword search, which aims at providing an overview of how fuzzy search scheme works over encrypted data. Assume Π=(Setup(1λ), Enc(sk, ·), Dec(sk, ·)) is a symmetric encryption scheme, where sk is a secret key, Setup(1λ) is the setup algorithm with security parameter λ, Enc(sk, ·) and Dec(sk, ·) are the encryption and decryption algorithms, respectively. Let Twi denote a trapdoor of keyword wi. Trapdoors of the keywords can be realized by applying a one-way function f, which is similar as [2, 4, 8]: Given a keyword wi and a secret key sk, we can compute the trapdoor of wi as Twi = f(sk,wi). The scheme of the fuzzy keyword search goes as follows: We begin by constructing the fuzzy keyword set Swi,d for each keyword wi ∈ W ($1 \leq i \leq p$) with edit distance d. The intuitive way to construct the fuzzy keyword set of wi is to enumerate all possible words w_ i that satisfy the similarity criteria ed(wi, w_ i) ≤ d, that is, all the words with edit distance d from wi are listed. For example, the following is the listing variants after a substitution operation on the first character of keyword CASTLE: {AASTLE, BASTLE, DASTLE, · · · , YASTLE, ZASTLE}. Based on the resulted fuzzy keyword sets, the fuzzy search over encrypted data is conducted as follows: 1) To build an index for wi, the data owner computes trapdoors Tw_ I = f(sk,w_ i) for each w_ I ∈ Swi,d with a secret key sk shared between data owner and authorized users. The data owner also encrypts FIDwi as Enc (sk, FIDwi _wi). The index table {({Tw_I}w_I∈Swi,d , Enc(sk,FIDwi_wi))}wi∈ W and encrypted data files are outsourced to the cloud server for storage; 2) To search with w, the authorized user computes the trapdoor Tw of w and sends it to the server; 3) Upon receiving the search request Tw, the server compares it with the index table and returns all the possible encrypted file identifiers {Enc(sk, FIDwi _wi)} according to the fuzzy keyword definition in section III-D. The user decrypts the returned results and retrieves relevant files of interest. This straightforward approach apparently provides fuzzy keyword search over the encrypted files while achieving search privacy using the technique of secure trapdoors. However, this approach has serious efficiency disadvantages. The simple enumeration method in constructing fuzzy keyword sets would introduce large storage complexities, which greatly affect the usability. Recall that in the definition of edit distance, substitution, deletion and insertion are three kinds of operations in computation of edit distance. The numbers of all similar words of wi satisfying ed(wi, w_ i) ≤ d for d = 1, 2 and 3 are approximately 2k × 26, 2k2 × 262, and 4 3k3 × 263,respectively. For example, assume there are 104 keywords in the file collection with average keyword length 10, d = 2,and the output length of hash function is 160 bits, then, the resulted storage cost for the index will be 30GB. Therefore, it brings forth the demand for fuzzy keyword sets with smaller size.

V. Constructions of Effective Fuzzy Keyword Search in Cloud

The key idea behind our secure fuzzy keyword search is two-fold: 1) building up fuzzy keyword sets that incorporate not only the exact keywords but also the ones differing slightly due to minor typos, format inconsistencies, etc.; 2) designing an efficient and secure searching approach for file retrieval based on the resulted fuzzy keyword sets.

A. Advanced Technique for Constructing Fuzzy Keyword Sets

To provide more practical and effective fuzzy keyword search constructions with regard to both storage and search efficiency, we now propose an advanced technique to improve the straightforward approach for constructing the fuzzy keyword set. Without loss of generality, we will focus on the case of edit distance d= 1 to elaborate the proposed advanced technique. For larger values of d, the reasoning is similar. Note that the technique is carefully designed in such a way that while suppressing the fuzzy keyword set, it will not affect the search correctness.

1. Wildcard-based Fuzzy Set Construction

In the above straightforward approach, all the variants of the keywords have to be listed even if an operation is performed at the same position. Based on the above observation, we proposed to use a wildcard to denote edit operations at the same position. The wildcard-based fuzzy set of wi with edit distance d is denoted as Swi,d={S_wi,0, S_ wi,1, · · · , S_ wi,d }, where S_ wi,τ denotes the set of words w_i with τ wildcards. Note each wildcard represents an edit operation on wi. For example, for the keyword CASTLE with the pre-set edit distance 1, its wildcard-based fuzzy keyword set can be constructed as:

S C A S T L E , 1 = { C A S T L E , * C A S T L E , * A S T L E , C*ASTLE,C*STLE, · · · , CASTL*E, CASTL*, CASTLE*}. The total number of variants on CASTLE constructed in this way is only 13 + 1, instead of 13 × 26 + 1 as in the above exhaustive enumeration approach when the edit distance is set to be 1. Generally, for a given keyword wi with length _, the size of Swi,1 will be only 2_ + 1 + 1, as compared to (2_ + 1) × 26 + 1 obtained in the straightforward approach. The larger the pre-set edit distance, the more storage overhead can be reduced: with the same setting of the example in the straightforward approach, the proposed technique can help reduce the storage of the index from 30GB to approximately 40MB. In case the edit distance is set to be 2 and 3, the size of Swi,2 and Swi,3 will be C1_+1+C1_· C1_+2C2_+2 andC1_ + C3_ + 2C2_ + 2C2 _· C1_ . In other words, the number is only O(_d) for the keyword with length _ and edit distance d.

B. The Efficient Fuzzy Keyword Search Scheme Based on the storage-efficient fuzzy keyword sets, we show how to construct an efficient and effective fuzzy keyword search scheme. The scheme of the fuzzy keyword search goes as follows:

1. To build an index for wi with edit distance d, the data owner first constructs a fuzzy keyword set Swi,d using the wildcard based technique. Then he computes trapdoor set {Tw_i } for each w_ I Swi,d with a secret key sk shared between data owner and authorized users. The data owner encrypts FIDwi as Enc (sk, FIDwi _wi). The index table {({Tw_ i}w_i ∈Swi,d , Enc(sk, FIDwi_wi))}wi∈ W and encrypted
data files are outsourced to the cloud server for storage;
2. To search with (w, k), the authorized user computes the trapdoor set {Tw_}w_∈ Sw,k, where Sw,k is also derived from the wildcard-based fuzzy set construction. He then sends {Tw_} w_∈ Sw,k to the server;
3. Upon receiving the search request {Tw_}w_∈ Sw,k , the Server compares them with the index table and returns all the possible encrypted file identifiers {Enc(sk, FIDwi _wi)}according to the fuzzy keyword definition in section III-D.The user decrypts the returned results and retrieves relevant files of interest. In this construction, the technique of constructing search request for w is the same as the construction of index for a keyword. As a result, the search request is a trapdoor set based on Sw,k, instead of a single trapdoor as in the Straightforward approach. In this way, the searching result correctness can be ensured.

VI. Security Analysis

In this section, we analyze the correctness and security of the proposed fuzzy keyword search scheme. At first, we show the correctness of the schemes in terms of two aspects, that is, completeness and soundness.

Theorem 1: The wildcard-based scheme satisfies both completeness and soundness. Specifically, upon receiving the request of w, all of the keywords {wi} will be returned if and only if ed(w,wi) ≤ k.

The proof of this Theorem can be reduced to the following Lemma:

Lemma 1: The intersection of the fuzzy sets Swi, d and Sw,k for wi and w is not empty if and only if ed(w,wi) ≤ k.

Proof: First, we show that Swi,d∩Sw,k is not empty when ed(w,wi) ≤ k. To prove this, it is enough to find an element in Swi, d ∩Sw, k. Let w = a1a2 · · · as and wi = b1b2 · · · bt, where all these ai and bj are single characters. After ed(w,wi) edit operations, w can be changed to wi according to the definition of edit distance. Let w = a* 1a* 2· · · a*
m, where a*i = aj or a*i =
if any operation is performed at this position. Since the edit operation is inverted, from wi, the same positions containing wildcard at w will be performed. Because ed(w,wi) ≤ k, w is included in both Swi,d and Sw,k, we get the result that Swi,d ∩ Sw,k is not empty.

Next, we prove that Swi,d∩Sw,k is empty if ed(w,wi) > k. The proof is given by reduction. Assume there exists an w belonging to Swi,d ∩ Sw,k. We will show that ed(w,wi) ≤ k which reaches a contradiction. First, from the assumption that w. Swi,d ∩ Sw,k, we can get the number of wildcard in w, which is denoted by n, is not greater than k. Next, we prove that ed(w,wi) ≤ n. We will prove the inequality with induction method. First, we prove it holds when n. = 1. There are nine cases should be considered: If w is derived from the operation of deletion from both wi and w, then, ed(wi, w) ≤ 1 because the other characters are the same except the character at the same position. If the operation is deletion from wi and substitution from w, we have ed(wi, w) ≤ 1 because they will be the same after at most one substitution from wi. The other cases can be analyzed in a similar way and are omitted. Now, assuming that it holds when n. = γ, we need to prove it also holds when n. = γ + 1. If ˆ w. = a1a2· · · a n∈ Swi,d ∩ Sw,k, where ai = aj or ai =a. For a wildcard at position t,cancel the underlying operations and revert it to the original characters in wi and w at this position. Assume two new elements w.i and w. are derived from them respectively. Then perform one operation at position t of w.i to make the character of w.i at this position be the same with w, which is denoted by w_i. After this operation, w.i will be changed to w, which has only k wildcards. Therefore, we have ed (w_i, w) ≤ γ from the assumption. We know that ed (w_i, w) ≤ γ and ed (w_i, wi) = 1, based on which we know that ed(wi, w) ≤ γ + 1. Thus, we can get ed (w,wi) ≤ n∈. It renders the contradiction ed (w,wi) ≤ k because n ≤ k. Therefore, Swi,d ∩ Sw,k is empty if ed(w,wi) > k.

Theorem 2: 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 in distinguish ability under the chosen keyword attack, which means there exists an algorithm A who can get the underlying information of keyword from the index.

Then, we build an algorithm A_ that utilizes A to determine whether some function f_(·) is a pseudo-random function such that f_(·) is equal to f(sk, ·) or a random function. A_ has an access to an oracle O f_(·) 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_(·). After making these trapdoor queries, the adversary outputs two challenge keywords w 0 and w1 with the same length and edit distance, which can be relaxed by adding some redundant trapdoors. A_ picks one random b {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, ·) or a random function. A_ sends y back to A, who answers with b_{0, 1}. Suppose A guesses b correctly with non eligible probability, which indicates that the value is not randomly computed. Then, A_ makes a decision that f_(·) is a
Pseudo-random function. As a result, based on the assumption of the in distinguish ability of the pseudo-random function from some real random function, A at most guesses b correctly with approximate probability 1/2. Thus, the search privacy is obtained.

VII. Conclusion

In this paper, for the first time we formalize and solve the problem of supporting efficient yet privacy-preserving fuzzy search for achieving effective utilization of remotely stored encrypted data in Cloud Computing. We design an advanced technique (i.e., wildcard-based technique) to construct the storage-efficient fuzzy keyword sets by exploiting a significant observation on the similarity metric of edit distance.

Based on the constructed fuzzy keyword sets, we further propose an efficient fuzzy keyword search scheme.Through rigorous security analysis, we show that our proposed solution is secure and privacy-preserving, while correctly realizing the goal of fuzzy keyword search

References

[1] R.Agrawal, J. Kiernan,"Watermarking Relational Databases", Proc. 28th Int'l Conf. Very Large Data Bases (VLDB '02), VLDB Endowment, pp. 155-166, 2002.

[2] P. Bonatti, S.D.C. Di Vimercati, P. Samarati,"An Algebra for Composing Access Control Policies", ACM Trans. Information and System Security, Vol. 5, No. 1, pp. 1-35, 2002.

[3] P. Buneman, S. Khanna, W.C. Tan,"Why and Where: A Characterization of Data Provenance", Proc. Eighth Int'l Conf.

[4] S. Jajodia, P. Samarati, M.L. Sapino, V.S. Subrahmanian, "Flexible Support for Multiple Access Control Policies", ACM Trans. Database Systems, Vol. 26, No. 2, pp. 214-260, 2001.

[5] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano, "Public key encryption with keyword search", in Proc. of EUROCRYP'04, 2004.

Mr. S. Durgaprasad received B.Tech. Degree in Computer Science Engineering from St.Ann's College of Engineering & Technology, Chirala, and Affiliated to JNTU Kakinada in 2010.He is currently pursuing M.Tech in Software Engineering at St.Ann's College of Engineering & Technology, which is affiliated under JNTU Kakinada.

Mr. VenkataNaryana received B.E. degree in Information Technology from Vignan Engineering College Guntur, in 2006, received M.Tech. Degree in Computer Science and Engineering from KL University, Vijayawada, in 2009, received M.Tech. He is currently working as Professor in Information Technology department in St.Ann's College of Engineering & Technology and totally he has 3 years of Experience.

Mr. J. Karthik received B.Tech. Degree in Computer Science Engineering from St.Ann's College of Engineering & Technology, Chirala, and Affiliated to JNTU Kakinada in 2010.He is currently pursuing M.Tech in Computer Science & Engineering at St.Ann's College of Engineering & Technology, which is affiliated under JNTU Kakinada. He published one National Level Conference Paper and his areas of interests are Data Mining & Data Warehousing, Software Engineering, Operating systems.