Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 46 (2015) 668 – 675

Procedia
Computer Science

International Conference on Information and Communication Technologies (ICICT 2014)

Aiding Effective Encrypted Document Manipulation Incorporated with Document Categorization Technique in Cloud

Samantha Susan Mathew* ,Hafsath C A*

*Dept. of CSE,Ilahia College of Engineering and Technology,Muvattupuzha,Kerala,India,686673

Abstract

Cloud computing involves storage as well as usage of computer resources and services online. It has emerged as a promising area for data outsourcing. Since the users data are stored in servers controlled by cloud service provider there are always concerns about the security of the stored data. The data can be encrypted to ensure its security but it will increase the complexity of the data access process. The presence of large amount of data in the cloud necessitates an efficient method for data access. An efficient keyword based search on the encrypted cloud data is proposed here which provides marked improvement over the traditional boolean search approach used in cloud environment. It uses the latest informational retrieval methods for the processing of the documents from the cloud. The homomorphic encryption technique is used for the building of a secure index which has the key role in the search process. The vector space model is employed for the representation of the text document as vector. Document categorization technique is incorporated for the easy maintenance and retrieval of the data from the cloud. It handles multiple data owners and users with a facility to store and retrieve multiple data files. The approach also makes sure that the processing on the user side is minimized while updating the existing document set.

Keywords: Homomorphic encryption; Vector-space model; Secure Searchable Index; Document Categorization

1. Introduction

Previously companies had to invest a lot for its IT infrastructure. The invention of cloud computing has helped the companies to reduce the investment in this regard and hence they can concentrate on the strategic projects. The main

* Corresponding Authors.(Samantha Susan Mathew). Tel.: +91-9496755904; (Hafsath C A.) Tel.: +91-9495317277;
E-mail address: susanmathew85@gmail.com ; *E-mail address:* hafsath.ca@gmail.com

doi:10.1016/j.procs.2015.02.120

advantage of cloud computing is the person who access the data needn't be in a specific location, the data can be accessed from anywhere provided proper authentication and authorization is present. Now a days more and more confidential information is moved to the cloud like the important government documents, personal emails, patients medical history, insurance data etc. The storage of these data in the centralized cloud helps the owner of the data be free from the burden of data storage while utilizing the maximum storage facility provided by the cloud. The key elements in a cloud computing scenario include the data user, data owner and cloud server. They don't reside in the same trustworthy area and hence maintaining confidentially of the data is a challenge. There are possibilities of data leakage from the server knowingly or unknowingly. The data is encrypted before outsourcing to protect the it. Encryption scrambles the data to make it unreadable and only authorized people can read it. In a cloud computing scenario, multiple data owners are present and they might be sharing large amount of data but the users might be interested only in explicit data files at a particular time.

In order to selectively retrieve the files keyword based information retrieval technique is used. This technique is commonly used in the plaintext context. The system will return only those files which contain the particular keywords present in the request. Unnecessary transmission of huge amount of data can be avoided by using this method. But in the cloud context the data is in encrypted form and performing the searching on such a format is a challenge. Fast and effective data access and storage is a pre-requisite in cloud and meanwhile the privacy and secrecy of data also has to be ensured. This paper proposes a method for selective search on the encrypted data present in cloud. The selection criteria may contain many keywords and the number of results required can be set by the requestor. Document categorization technique has been incorporated to it to make the data storage and retrieval in a more arranged fashion and provides a better user experience. This enhances the user interface as the users can view the data files based on category.

The proposed system ensures the utilization of the high computational capability of the server while preserving the confidentiality of the data. On the server the scoring is performed and on the client the rank calculation is done. For creating the index for search and also the search request the idea of similarity relevance, vector space model and homomorphic encryption are used.

The file collection of an owner can be updated effortlessly by accessing the existing encrypted score values of the existing file collection. This reduces the burden of reparsing the existing file set of the owner. The file that has to be removed or uploaded is parsed again and the existing index is updated with that information to create the new index which is outsourced to cloud. The communication overhead can also be reduced as the existing files need not be retransmitted to server.

2. Related Works

Much research works has been done on how to search over encrypted cloud data securely and efficiently. Applying a searchable encryption in cloud computing addresses this problem to an extent.

Searchable encryption is a technique that allows users to search over encrypted data using keywords without decrypting it. The two main categories of searchable encryption are symmetric-key version and public-key version. Song et al. [1] proposes a searchable encryption scheme which uses the symmetric-key version. Some techniques for searching over encrypted data with data confidentiality are proposed. This approach has high computational complexity during the search operation. In commercial cloud this construction may not provide accurate result, as they are developed as crypto primitives. A method called Public Key Encryption with keyword Search (PEKS) [2] is proposed by D. Boneh, which handles the issue of searching in encrypted data. It uses the public key form or asymmetric form of searchable encryption. This is mainly applied in a mail gateway to filter the mails based on pre-defined keywords. These keywords are encrypted using PEKS. The mail gateway will not learn anything about the encrypted mail and hence the user privacy is ensured. The trapdoor is also encrypted with PEKS and the match is checked by using a test function. Here the sender of the mail has to specify the keywords.

A boolean search is carried out in the legacy keyword search which only identifies the occurrence of a particular keyword. A ranking technique based on Order Preserving Mapping is proposed which protect sensitive score information[3]. It flattens the original relevance score distribution which will increase its randomness but preserves the plaintext order. This method provides good data relevance based on term frequency. It supports only single keyword based top-k retrieval. A privacy-preserving multi-keyword ranked search method is proposed by Cao et al. which performs a ranked search on encrypted data[4]. Here the co-ordinate matching method calculates the rank using number of matched keywords. Inner product similarity and co-ordinate matching calculates the similarity between the documents and query. It performs a multi-keyword search but ranking is based only on the number of retrieved keywords.

A method for accessing documents from encrypted cloud data using a Confidential Index is present[5]. It performs top-k retrieval from confidential outsourced inverted index. The relevance score of a term q in a document d is as rs (q,d)=$\frac{\mathrm{TFq}}{|\mathrm{d}|}$ where TFq is number of times a particular keyword q appears in document d and |d| is the document length. To improve the security of the indexed data, a relevance score transformation function (RSTF) is used to make the relevance scores of different terms indistinguishable. The client may send a follow up request if desired number of elements is not received. Here the interdependency between the files is not considered.

A framework for rank-ordered search and retrieval over large document collection is proposed[6].It handles the searching of the document by the content owner and the searching in the data centre by people other than the content owners separately. An order preserving encryption is used as an inner layer of encryption which is applicable to the non-content owners and an outer layer of encryption which is applicable for everyone using the system. This encryption is used to encrypt the Term Frequency (TF) values. The Inner Layer Encryption performs computations and ranking directly on term frequency data in its encrypted form which is an order preserving encryption. An efficient solution using a secure traversal framework and an encryption scheme based on privacy homomorphism is proposed[7]. This method can be used in large datasets and efficiency of the query processing is high. It doesn't support atop- queries, skyline queries and multi-way joins. The main issues it faces are the Boolean representation and maintaining the balance between security and efficiency.

A method which supports Boolean based keyword retrieval is proposed where the user wants to search for a particular document provides a capability for the word W which is provided to the server[8]. The server identifies the documents that satisfy the user's request. Here the capability is generated based on the fields and hence mainly applicable for searching in the encrypted emails. Searchable symmetric encryption (SSE) is a cryptographic technique that allows a user to outsource the storage of its data to a server in a confidential way, while maintaining the ability for keyword-based search over it[9]. While most of the system is based on a single user setting describes a procedure that can be applied on a multiuser environment. It constructs schemes provably secure against non-adaptive and adaptive adversaries. The method is not dynamically scalable.

3. Proposed System

The three key elements involved in a cloud computing ecosystem are the owner of the data, the user who access the data and the cloud server provided by the cloud service provider. The set of files that needs to be outsourced to the cloud server is owned by the data owner. The data owner's files will be encrypted and stored in the cloud server to provide additional security. The data owner authorizes the set of users who can access their data present in the cloud server. The suggested method ensures that a multi-keyword search can be performed by any data users on the set of files uploaded to the cloud by the data owners. The above mentioned search requests will return a set of relevant documents based on category whose limit can be set by the requester.

3.1 Data Owner

In a cloud computing paradigm multiple data owners are present and each one has a collection of files to outsource to the cloud. Let $C_F=\{f_1,f_2,\ldots f_n\}$ be the set of files of a data owner. An index corresponding to the set of files are maintained for making the search procedure simple, fast and efficient. The index is generated by extracting the list of keywords from the file collection C_F. From the keyword list the stop words and very less frequently occurring words are discarded. This helps to reduce the size of the index structure. Further reduction of the index size is achieved by applying stemming on these word set. Let the so generated word list is, WL= $\{w_1,w_2…w_l\}$ where l is the number of keywords present in the list.

3.1.1. Document Categorization

During the process of uploading a file to the cloud server, a provision is provided for the data owner to specify the categories it belongs to. The word list WL will be generated for each of these categories specified by the data owner. It helps in displaying the results based on category which would make the search more relevant to the data user.
This also ensures that during any update operation of the file set the index of the corresponding categories alone needs to be updated. This results in significant savings by reducing the processing overhead.

3.1.2 Index Building

The scores for each keyword, w_i (1<=i<=l) in the wordlist WL are calculated. The relevance can be found out from the score values. The scores are calculated using the tf-idf weighing mechanism. This scheme considers two attributes – the term frequency (tf) and inverse document frequency(idf). The number of occurrence of a word w in a file f is denoted by term frequency (tf). Because of the variable length of the documents, some words may appear much more times in long documents than shorter ones. In order to reduce its effect, the term frequency is often divided by the document length N. Thus TF(w) = (Number of times word w appears in a document) / (Total number of words present in the document). The importance of a word is measured using the Inverse document frequency. It will weigh down the frequent terms while scale up the rare ones. IDF(w) = log(Total number of documents / document frequency). Document frequency defines the total number of files that contains the particular word w. The weight of each word w in the wordlist WL with respect to a file f is $\text{tf-idf}_{w,f} = \text{tf}_{w,f} * \text{idf}_w$. In the index building procedure, for each file in the file set of a category the score of each word in the wordlist are calculated. This depicts the weight of the particular keyword on the file in that category.

A vector space model is used to represent the score of the files on multiple keywords. It is an algebraic model for representing the text documents as vectors. The number of dimension corresponds to the number of keywords and the number of files considered. The vector will have the tf-idf value for each word that occurs in the document. For a word that doesn't occur in the document the corresponding tf-idf value will be zero. So for each file f_i in a category the data owner builds a (l+2) dimensional vector $d_i=\{id_i,ca_i,t_{i,1},t_{i,2},…t_{i,l}\}$ where l is the length of the word list ,id_i is the identifier of the file, ca_i is the category to which the file belongs to and $t_{i,j}=\text{tf-idf}_{wj,fi}$. The searchable index for a category SI=$\{d_i|1\leq i \leq N)$ where N is the total number of document in that category.

To make the searchable index secure it is encrypted using a special type of encryption called homomorphic encryption. Homomorphic encryption is a form of encryption which allows specific types of computations to be carried out on ciphertext and generate an encrypted result which, when decrypted, matches the result of operations performed on the plaintext. In the setup phase of homomorphic encryption the data owner generated a secret key SK' and a set of public keys PK'. These will be shared with authorized data users. The secure searchable index is created by encrypting each value in the searchable index by any of the key from the public key set ie SI'=$\{d'_i|1 \leqslant i \leqslant N\}$, where $d'_i = \{id'_i,ca_i',t'_{i,1}, t'_{i,2}, . . . , t'_{i,l}\}$, id'_i =Encrypt($P_{i,0}$, id_i) , ca'_i =Encrypt($P_{i,1}$, ca_i) and $t_{i,j}$'= Encrypt($P_{i,j}$, $t_{i,j}$) ($P_{i,0} \in$ PK', Pi,j $\in$PK' ,1 $\leqslant$ j $\leqslant$ l). The above approach helps in preventing the access pattern and search pattern analysis by the attackers by making sure that different keys from the public key set are used for the generation of the encrypted score values.

3.1.3 Data Outsourcing

Once the secure index is generated, the data owner needs to make sure that the file collection is also encrypted before outsourcing it to the cloud server. Any general encryption method can be used for the same. We used a type of symmetric encryption called AES (Advanced Encryption Standard) for encrypting the files. The encrypted file collection and the secure index of each category will be outsourced to the cloud server.

3.1.4 Data Modification

Frequent changes to the file collection of a particular data owner are very common in real life cloud computing scenario. An efficient and reliable mechanism which won't burden the user side is desirable in this case. The proposed approach facilitates the data owner to add more documents into his existing file collection as well as the deletion of his already uploaded documents. The system will automatically take care of any changes required to the index there by relieving user from the index maintenance.

The data owner is generating the private key and public key for the secure index structure creation. Hence for any of the above mentioned operation, the index is retrieved from the cloud server based on the categories specified by the data owner and decrypts the same. Hence without parsing or calculating the word list, term frequency, inverse document frequency and tf-idf values of the words in any of the existing file collections, these values can be found out from the decrypted index structure. The data owner finds out the word list of the new documents, appends the new words to the existing word list WL of the corresponding category, calculates new tf-idf values, encrypts the index structure and outsource it to cloud.

Similar set of operations will be performed for the file deletion from the set of files owned by the data owner. The secure searchable index SI' is decrypted by the data owner, finds out the words in the word list that corresponds to the file that has to be deleted and updates the index structure accordingly.

The Fig. 1 illustrates the working of the data owner module.

.

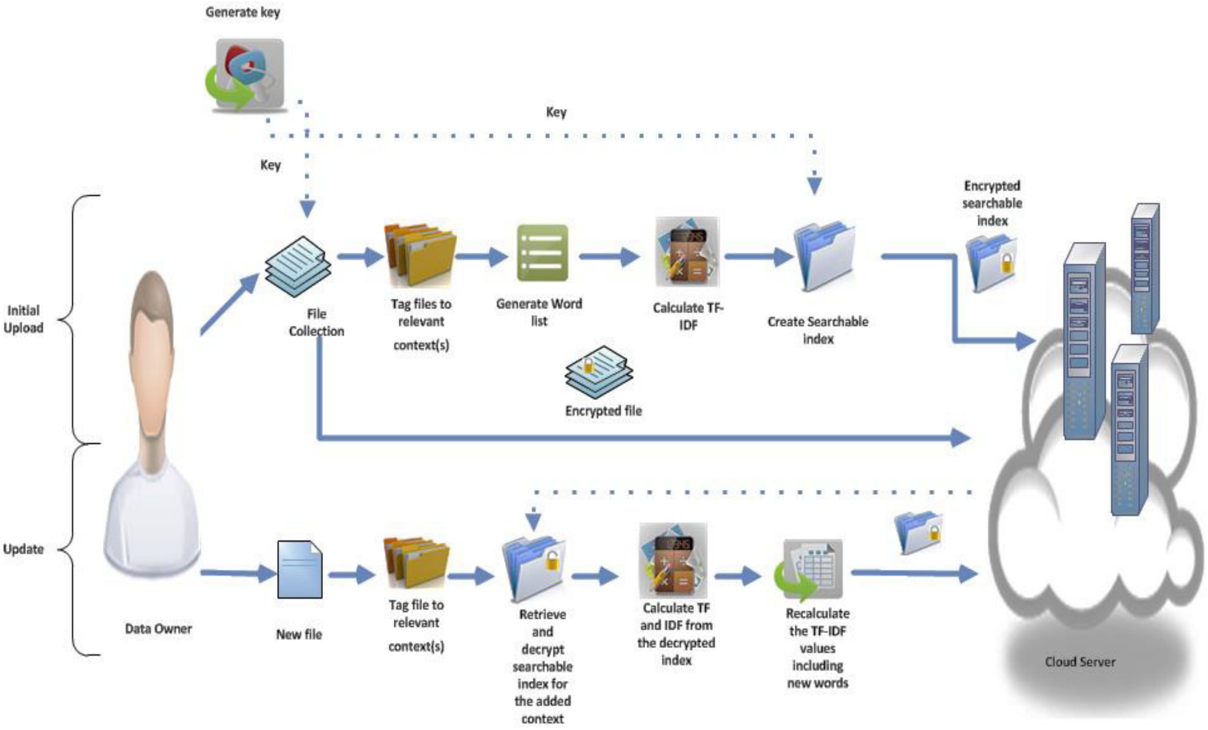
Generate key
Key
Key
Initial Upload
Update
Data Owner
File Collection
Tag files to relevant context(s)
Generate Word list
Calculate TF-IDF
Create Searchable index
Encrypted searchable index
Encrypted file
New file
Tag file to relevant context(s)
Retrieve and decrypt searchable index for the added context
Calculate TF and IDF from the decrypted index
Recalculate the TF-IDF values including new words
Cloud Server

Fig. 1 Actions performed by data owner

3.2 Data User

The outsourced encrypted data in the cloud can be accessed by the data users who are authorized by the data owner. The identifiers of the document along with the score value are send back to the user. The ranking operation i.e. a sorting of the score value in descending order is performed at the data user side. The ranking is performed on the user side prevents leakage of sensitive information that may usually occur in the case of server side ranking.

3.2.1 Request Generation

During the search operation performed by the data user, multiple query vectors are generated (one for each category) based on the keywords provided by him. It is assumed that the word list WL, of the categories corresponds to each data owner who has shared the access to a particular data user is transferred to him in a secure fashion. A query vector is generated for each of these categories. For the creation of the query vector the binary values 0 or 1 is used depends on the appearance of the requested keyword in the securely transferred wordlist WL. The trapdoors are encrypted using the set of public keys present in the user side. Suppose that the requested set of keywords be REQ $=\{w_1', w_2', . . . , w_s'\}$, then generated query vector is of the form TD= $\{t_1,t_2, . . .,t_l\}$ where $t_i =1$ $(1 \leqslant i \leqslant l)$ if $t_i \in$ REQ and $t_i = 0$ otherwise. This created trapdoor is encrypted to form a secure trapdoor TD' = $\{t_1', t_2', . . . , t_l'\}$, where t_i' =Encrypt(R,t_i) and R ⊆ PK'. This secure trapdoor is forwarded to the cloud server.

3.2.1 Reply Processing

The response from the cloud server is the result vector RV=$\{(id_1',ca_1',p_1'),(id_2',ca_2',p_2'),….(id_n',ca_n',p_n'))\}$ where id_i' is the identifier of the file i, ca_i' is the category corresponds to the file i and p_i' is its score relevant to the query vector. The result vector will be in encrypted format. The result vector can be decrypted by using the secrete key to get the scores of each file. This is followed by the ranking procedure which group the score based on category and sort it in descending order. The user can determine the number of files in the result set and if it is k', he sends the identifiers of the top k' files to the cloud server. The encrypted files correspond to the requested set of file identifiers are forwarded by the cloud server to the user. These files are decrypted by the data user and can be used as required..

The Fig. 2 illustrates the working of the data user module.

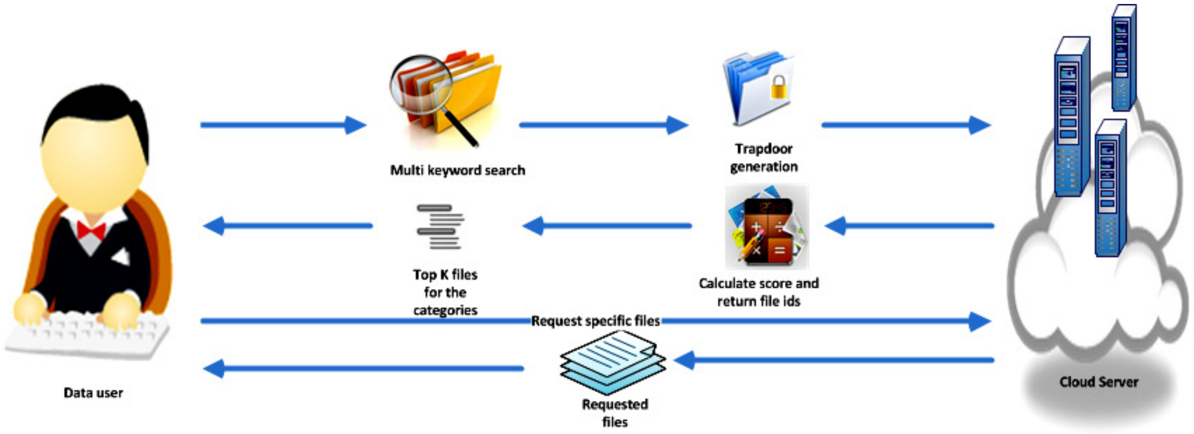

Fig. 2 Actions performed by data user

3.3 Cloud Server

Since the data files as well as the tf-idf values are encrypted before uploading it to the cloud server, the cloud server will have minimal information about the files stored as well as about the user queries. As the server is having high computational capability than the client we will be performing most of the CPU intensive processes in the cloud server. On receiving the request vector(s) from the authorized data user, the cloud server will perform an inner product operation using the request vector and encrypted tf-idf values to obtain the relevant score for each of the categories. These operations will be restricted to the indexes of data owners who shared the data with the requested data user. With homomorphic encryption in place, the computation can be performed on the encrypted data similar to what can be done on the plain text. The decrypted value of the result will be same as what would obtain if the same operation is applied on the plain text. This property of homomorphic encryption ensures that there is no information leakage in the server. The score values at each category level will be send back to user in encrypted format which will be decrypted and sorted later at user side. This ensures that the processing is minimal at the user side.

4. Solution Methodology

The index of the document collection uploaded by the data owner is made secure by encrypting it with an encryption method called homomorphic encryption [10]. This main steps involved in this encryption methodology are: KeyGen, Encrypt, Evaluate, and Decrypt.

KeyGen(μ): The KeyGen procedure generates the secret key and the group of public key. The secret key SK' is an odd ɱ-bit number from interval $[2^{ɱ-1},2^{ɱ})$. The set of public keys $PK' = \{k_0,k_1,\ldots k_{Ʈ}\} \subseteq \{pq + xr,\ q \in [0, 2^{\gamma}/p)$, r $\in(-2^{ƥ},2^{ƥ})\}$ where ɱ-bit-length of the secret key , $ɱ = \Theta(\mu^2)$, Ʈ -number of integers in the public key, p-secret key, q- a multiple parameter, r-noise to achieve proximity against brute force attack. The noise parameter x is considered as $x = 2^{2||m||}$ where $||m||$ is the bit length of the cipher text.
Encrypt(PK',m): The Encrypt() function generates takes the plain text and the group of public keys to produce the cipher text as , c=pq+xr+m
Evaluate(c1,c2,…ct): Repeated addition and multiplication can be applied on the input cipher text set and a integer X is produced as output.
Decrypt(SK',X): The decrypt function produces plain text as output m'=(X mod SK') mod x.

The overall process involved in the multi-keyword based search can be divided into an setup phase, modification phase and data access phase.
Setup Phase:

1. Data owner generate the secret key and the public key set using the KeyGen(μ) function in homomorphic encryption and share the same to authorized users.
2. For each category of files, data owner extracts the set of keywords (say l), stem it and finds the TF-IDF values of those words. For each file an (l+2) dimensional vector is generated using the calculated TF-IDF, file id and category id values to form the searchable index
3. Construct the secure searchable index using the public key set and upload the same to cloud server along with encrypted file collection

Modification Phase:

1. Identify the category of the new file to be added or deleted and retrieve the secure searchable index
2. Decrypt using the secret key and find out the old tf-idf values
3. Recalculate the tf-idf values based on the change and recreate the searchable index.
4. Encrypt it again and upload to the cloud server

Data access Phase:

1. Based on the keywords in the search request a query vector is generated for each category by putting 1 if the term in the word list is present in the request otherwise mark it as zero.
2. An inner product of the encrypted query vector sent to the cloud server and secure searchable index is calculated for each category to form the result vector containing the id of the files, the category and the encrypted score value which is send to the user.

3. The result vector is decrypted using the secrete key at user side and ranking is performed for each category. The top-k highest scoring file identifiers are send to the cloud server and corresponding files will be send back by the cloud server.

5. Conclusion

This paper resolves some of the key problems of multi-keyword search over encrypted data by ensuring confidentiality of the data with higher performance. The focus of the approach is to perform the bulk of the processing at the cloud server side which has high computation power. The confidentiality is ensured by using homomorphic encryption technique. The categorization of data as well as the new method for updating the index by retrieving the required attributes from the existing searchable index helps in reducing most of the overhead associated with the current methods for updation. The size of the cipher text and the key length is high in the current implementation. With the recent works of the cryptography community towards the implementation of a more practical full homomorphic encryption the efficiency of the proposed method can be further enhanced.

Acknowledgements

The authors wish to thank the Management, Principal, Head of the Department and Staff members (CSE) of Ilahia College of Engineering and Technology for their support and help in completing this work.

References

1 D. Song, D. Wagner, and A. Perrig, Practical Techniques for Searches on Encrypted Data, Proc. IEEE Symp. Security and Privacy, 2000.

2 D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, Public- Key Encryption with Keyword Search, Proc. Int'l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt), 2004

3 C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, Secure Ranked Keyword Search over Encrypted Cloud Data, Proc. IEEE 30th Int'l Conf. Distributed Computing Systems (ICDCS), 2010.

4 N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, Privacy-Preserving Multikeyword Ranked Search over Encrypted Cloud Data, Proc.IEEE INFOCOM, 2011.

5 S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, Zerber+r: Top-k Retrieval from a Confidential Index, Proc. 12th Int'l Conf. Extending Database Technology: Advances in Database Technology(EDBT), 2009.

6 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, Proc. Workshop Storage Security and Survivability, 2007.

7 H. Hu, J. Xu, C. Ren, and B. Choi, Processing Private Queries over Untrusted Data Cloud through Privacy Homomorphism, Proc. IEEE 27th Int'l Conf. Data Eng. (ICDE), 2011.

8 P. Golle, J. Staddon, and B. Waters, Secure Conjunctive Keyword Search over Encrypted Data, Proc. Second Int'l Conf. Applied Cryptography and Network Security (ACNS), p. 31-45, 2004.

9 R. Curtmola, J.A. Garay, S. Kamara, and R. Ostrovsky, Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions, Proc. ACM 13th Conf. Computer and Comm. Security (CCS), 2006.

10 M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, Fully Homomorphic Encryption over the Integers, Proc. 29th Ann. Int'l Conf. Theory and Applications of Cryptographic Techniques, H. Gilbert,p. 24-43, 2010.

11 Yu, Member, IEEE, Peng Lu, Yanmin Zhu, Member, IEEE, Guangtao Xue, Member, IEEE Computer Society, and Minglu Li , Toward Secure Multikeyword Top-k Retrieval over Encrypted Cloud Data. Jiadi