International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 09, September-2015

Efficient Semantic Keyword Search and Dynamic Update over Encrypted Cloud Data

Meenu K. R.
PG Scholar, Dept. of CSE,
Ilahia College of Engineering and Technology
Muvatupuzha, Kerala, India

Theresa Jose
Asst. Professor, Dept. of CSE,
Ilahia College of Engineering and Technology
Muvatupuzha, Kerala, India

***Abstract*— The cloud computing paradigm motivates many of the data owners to store their confidential data in cloud for greater sharing and storage services. The risk of storing confidential data in the public cloud forced the data owners to store data in encrypted form. Thus, the search over encrypted data is of predominant importance to provide maximum data utilization for encrypted cloud data. The privacy of that data is controlled by the cloud service provider which demands for strict data privacy protection. The traditional privacy- preserving Multi-keyword Ranked Search over Encrypted Cloud data (MRSE) schemes involves much overhead in terms of storage and search time. The proposed system solve the problem of MRSE with less computational and storage overhead. Also facilitates dynamic update of the encrypted data and privacy of the data is also maintained. The system also includes a semantic search facility can improve the search experience by utilizing the NLP library. Thorough analysis investigating the efficiency of the scheme is given which guarantees that there is less overhead involved compared to the previous MRSE schemes.**

Keywords*— Cloud computing; multi-keyword ranked search; semantic search; dynamic update; NLP Library

I. INTRODUCTION

Many organizations are prompted to outsource their computing and storage needs with the increasing needs of computing resources which is commonly referred to as cloud computing. To enjoy the high-quality on-demand services and applications from a common pool of resources, the cloud customers can store their data in cloud remotely. But, there is great risk in storing the confidential data as it is in the public cloud. Thus the data owners are forced to encrypt their data before upload. But, this should not affect the data utilization mainly being the document search. It is not affordable to download the whole document set and then apply the traditional keyword search on plaintext. So a search technique which can be applied over encrypted data is of predominant importance. The coarse results obtained with a single keyword search in most of the existing system can be made efficient by facilitating a multi-keyword search. With multiple keywords, the user can specify the required files more specifically. A ranked search enables the users to get the most relevant result that matches the query in the "pay-as-you-use" cloud paradigm. Only the most relevant data is retrieved. Thus a multi-keyword ranked search can make the searching over encrypted data very efficient. While considering the relevance ranking we ensure to consider the importance of each term in the particular document as well as the whole set of documents. That means, term frequency (TF) and inverse document frequency (IDF) contributes for better result.

Searching can be made fine grained with better search queries and more accurate if the search system is efficient enough to understand the intend of the user. A semantic keyword search is thus essential.

Encrypted index based search techniques are prominent. Cloud System is known as "honest but curious" system. That means it is curious to know about the documents stored in the cloud server. It always try to learn about the contents even in encrypted form in the server. Also, even the encrypted data in cloud server can be can accessed by the public. So privacy preserving search which prevents even the search keyword exposure demands much attention.

"Coordinate matching" [10] as many matches as possible is a good similarity measure in the plain text information retrieval (IR) for multi-keyword search with refined result relevance. But in encrypted cloud data this cannot be applied. We need to consider the data privacy, index privacy, query privacy etc.

In this scenario there is a data owner who uploads the documents to the cloud server and assign some sharers to his documents. A sharer is an authorized user who is able to perform search and access the documents being shared to him. It is required to provide a user environment that allows multiple users to upload and provide search for all the documents to which the user has access. The MRSE schemes in [9] supports sharing the whole set of documents owned by a user. It is necessary to facilitate a flexible sharing, where owner can share subset of documents to his sharers.

To perform search on a document collection with some keywords, authorized users acquires a corresponding trapdoor through search control mechanisms. It is necessary to maintain better search control and access control. Search control regulates how authorized users acquire trapdoors. Access control manages the user's access to outsourced documents.

The system scalability, usability and performance requirements become challenging in this scenario of increasing on-demand data users and huge data collection being outsourced. Hence a faster mechanism with minimum overhead in computation and storage in the "pay-as-you-use" cloud paradigm for the privacy-preserving MRSE is of paramount importance.

The proposed system performs semantic keyword search over encrypted cloud data with maximum privacy settings and

minimum overhead. Also maintain a better access control and search control with a trust server to manage all related keys and metadata about the encrypted data as well as the index for searching. A user environment supporting storage of data from multiple data owners and search from multiple users is also set up. The system also supports dynamic update of the documents uploaded by several users.

II. RELEATED WORKS

Related to the encrypted data search in cloud paradigm and privacy settings related to this, many research works are in progress.

Song et al. [1] for the first time proposed the searchable encryption scheme with symmetric setting. Different techniques were proposed for searching in the encrypted data while providing the provable secrecy, controlled access and hidden queries. But, the proposed methods could not deal with compressed data, also it supported only single keyword search and the search could easily learn the search patterns.

E. J. Goh [2] proposed secure indexes using the pseudo random generators and bloom filters. But the bloom filters induced false positives which was not suited in the cloud paradigm which causes users to download unwanted documents and pay for that.

D. Boneh, proposed a method [3] known as Public Key Encryption with keyword Search (PEKS), which performs searching in encrypted data by using asymmetric version of searchable encryption. The owner with a public key can store to the server but the user with a private key only can search. But the method supports only a few number of keywords and only a boolean keyword search can be performed. The method is computationally expensive and consider only the problem of retrieving e-mails from a user's e-mail server which contains a particular keyword.

SSE, Searchable symmetric encryption [4] is a cryptographic technique that allows a user to outsource the storage of its data to a server in a confidential manner, and at the same time maintaining the facility for a keyword-based search over it. SSE schemes can be constructed by combining a secure index and a symmetric encryption scheme. A multiuser setting was done where not only data owner can search. But only search with single keyword was possible.

Multi-keyword searches were introduced in [5],[6] , but did not provide a ranked result as well as the privacy requirements.

In [7] E. Shen et al. proposed the ranked search concept. Use the statistical measure approach, i.e. relevance score to build a secure searchable index. Facilitates an efficient server-side ranking without losing keyword privacy. Embedded a weight information of each file during establishment of searchable index before outsourcing the encrypted files which was calculated by TF-IDF rule.

W.K. Wong et al. in [8] proposed Secure Computation ON an Encrypted DataBase called SCONEDB which solved the problem of computation of kNN (k nearest neighbor) on an encrypted database. A new model ASPE, asymmetric scalar-product-preserving encryption is used to select k nearest database records. Euclidean distance between a data record pi and a query vector q is considered. The method introduced randomness in the encrypted search index and the query. But the cloud server could learn relations about the index and query.

Ning Cao et al. in [9] used the inner product similarity at the same time providing much more privacy than the secure kNN method by inserting dummy keywords in the query and searchable index. But insertion of dummy keywords incurs unnecessary cloud storage space which in turn is not suitable for the "pay-as-you-use" cloud paradigm. Much computational overhead is induced during the scalar inner product for actual search. The same overhead is induced for the dummy keywords also which is not affordable.

In [11] Sun et. al. brought out schemes that support secure mode of semantic search with semantic extension of the query keyword. A semantic relationship library is constructed based on the probability that certain keywords co-occur in the document set. However, the scheme considers only the co-occurrence of the terms for semantic relationship. But the co-occurrence within a small data collection is not sufficient to get the exact semantics or intend of the search user.

Existing systems concentrate on the privacy settings neglecting the overhead induced. So a search technique which induces less overhead and thereby increase the search speed is very essential. A semantic search which is capable of capturing more semantics than co-occurrence relationship is of paramount importance. Most of the existing system do not consider search control and access control.

III. PROBLEM FORMULATION

A. Existing System Model

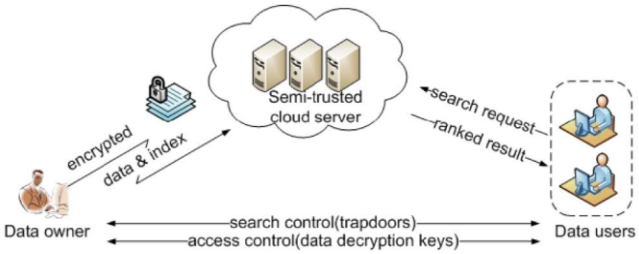

Fig. 1. Existing architecture of the search over encrypted cloud data

Consider a cloud storage system with the cloud server, a data owner and data user as illustrated in Fig .1 [9].The data owner outsources a set of document set F to the cloud server in an encrypted form C. For effective search over C, the data owner build a searchable index I from set of documents F and encrypt it to E before outsourcing. The indexes are constructed on a user level i.e. for each set of documents uploaded by a particular data owner. The encrypted index E and the encrypted set C is then outsourced to the cloud server. Only authorized users who acquires a trapdoor T from the data owner to can perform search for t keywords on the encrypted set C. When cloud storage server receives search request from a data user, the cloud server performs a search on E, and return the corresponding ranked results. Relevance ranking based on the TF-IDF is provided and privacy settings are also provided which will be introduced shortly. To reduce the communication cost, a parameter k is also accepted from the user for retrieving the top-k documents.

B. *Proposed System Model*

Consider Fig. 2 for the architecture of the proposed system.

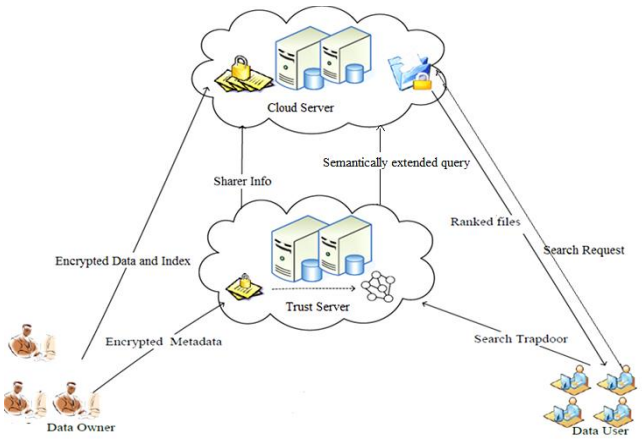

Fig. 2. Proposed architecture of the efficient semantic search over encrypted cloud data

The proposed model includes a trust server which provide the support for access control and search control. The trust server is maintained at the client side or the organization side for the better exchange of the keys and maintenance of metadata about the encrypted data stored in cloud server. Multiple users can upload the documents and search is enabled for multiple users on all the documents to which they have access. In the existing system the keys for generating the trapdoor T is passed through some secure method but once it is received by the user it is difficult to store it secretly. The chance for a leak of the key is much higher in such case. Trust server placed in between the data owner and the data user solves this leakage to a great extend. A validation is performed at the cloud server side also, so that a key leakage at the client side alone cannot lead to an attack by an external user i.e. cloud server also maintains an information about the sharer.

An encrypted hashed index I is created and for each document. The encrypted data C along with index is then kept in the cloud server. The search user who is authorized creates a search trapdoor and is semantically extended to create a semantically extended query, SQ to enable the semantic search. On receiving SQ from the authorized user, cloud server ranks each document based on the SQ and sends the top-k ranked results back to the user.

C. *Security Model*

The cloud system is considered as 'honest-but-curious' system. It is honest in the sense to the protocol specifications. But it is curious to know about storage, index and the message flows during protocol. It keep on trying to get additional information. Depending on what cloud server knows, two threat models are proposed.

Known ciphertext model: In this model, the cloud server has a basic knowledge about the encrypted data set and encrypted index.

Known background model: This model is much stronger in which the cloud server knows extra background information related to the information stored in it. It may include trapdoor information or some statistical information including the document/term frequency.

Since cloud server is available to public, the probability of an attack by an external user is more. So the secured search is very important.

The proposed scheme ensures i) document privacy which is achieved by encrypting the content before outsourcing ii) index privacy which is achieved by encrypting the index created for search and iii) search privacy which is much harder to achieve where the keywords are not expected to be exposed in the cloud in any way, either in index or in query.

D. *Existing MRSE Framework*

In [9], Ning Cao et al. describes the basic MRSE scheme, MRSE_I, MRSE_II, MRSE_I_TF and MRSE_II_TF schemes. The MRSE_II_TF framework is discussed below. The following steps are involved.

- **Setup (l,λ)**: An n dimensional dictionary vector $W=(W_1, W_2,..., W_n)$ is constructed for each document set $F=(F_1, F_2, ..., F_m)$ uploaded by the data owner, where n represents the total number of unique terms in F and m represents the number of documents. The data owner creates a symmetric key SK with a security parameter *l* as input. SK is composed of $[S,M_1,M_2]$. S is an (n+U+1) dimensional split vector where U denotes the number of dummy keywords inserted. M_1 and M_2 are two (n+U+1)X(n+U+1) dimensional invertible matrices. The dimension extension with the dummy keywords are insert is to induce more randomness so that the cloud server get confused. Each document set F is encrypted as $C=(C_1, C_2,... ,C_m)$.
- **BulidIndex(F, SK)**: Based on dictionary W index $I=(I_1, I_2,...I_m)$is constructed. For each F_i, subindex I_i is constructed such that each binary bit $I_i[j]$ represents whether the corresponding keyword $W_i[j]$ is present in the document. The TF-IDF value represents the importance of each term in the document set F. The corresponding value is calculated by the TF*IDF rule [9] and stored in the index position $I_i[j]$ of I_i. Then each vector I_i is split into 2 vectors I_i' and I_i''.The following rule is used for splitting. $I_i'[j]= I_i''[j]= I_i[j]$ if s_jЄ S is 1. Else $I_i'[j]= \frac{1}{2} I_i[j] + r$ and $I_i''[j]= \frac{1}{2} I_i'[j] - r$ where r is any random number. Finally, the encrypted subindex $E_i = \{M_1I_i', M_2I_i''\}$ is built for each C_i.
- **Trapdoor**: On a query processing with keywords set w for search in W as input, a binary vector Q of dimension (n+U+1) is generated. Each bit Q[j] indicates whether presence of keyword at $W_i[j]$ is true or false. Vector Q is split to Q' and Q" similar to splitting of I. The Trapdoor ,T is generated as $\{M_1^{-1}Q', M_2^{-1}Q''\}$.
- **Query**: On receiving trapdoor T, the cloud server computes the relevance score for each document related to the query as

$$Score= \{M_1I_i', M_2I_i''\}.\{M_1^{-1}Q', M_2^{-1}Q''\} \quad (1)$$

The existing system has proposed the insertion of dummy keywords for confusing the cloud server. But there is a wastage of cloud storage in the "pay-as-you-use" cloud paradigm. For a trapdoor generation the whole unique keyword list or the dictionary has to be shared to the search user. The maintenance of a matrices and multiplication during BuildIndex step incurs much computational overhead. For the dummy keywords also

the computations are done which add to the normal overhead. During trapdoor generation the matrix multiplication is done for the non query keywords also. So faster and secure search mechanism is of predominant importance.

E. Design Goals

- *Multi-Keyword Ranked Search*: The scheme for providing search with multiple keywords and provide results which are ranked based on their relevance.
- *Privacy Preserving*: To maintain the strict privacy requirements even in the known background model.
- *Search Control:* To regulate how authorized users acquire trapdoors.
- *Access Control:* To manage user's access to outsourced documents.
- *Semantic Search:* To perform search that includes user intend.
- *Dynamic update:* To facilitate update operation on the already uploaded documents updating the TF-IDF values efficiently.
- *Efficient Search:* To facilitate a search that induces less computational and storage overhead.
- *Multi-user environment*: A user environment supporting storage of data from multiple data owners and search from multiple users.

IV. SOLUTION METHODOLOGY

Architecture of the advanced scheme of efficient semantic search over encrypted cloud data is shown in the Fig. 2.

A. Efficient Semantic Search over Encrypted Cloud Data

The scheme has the following basic steps.

- **Setup:** When a user U uploads document set $\{D_1,D_2,...,D_d\}$, for each document D_i the unique terms $term\_id_{ij}$ are extracted. Checksum or unique hash value, CS_{ij} is calculated for each $[file\_id_i, term\_id_{ij}]$ pair such that $1<i<d$ and $1<j<t_i$ where d is the number of documents and t_i is the number of terms in D_i. Each checksum CS_{ij} is encrypted as $Enc(CS_{ij})$ is created. Frequency of each term is also counted and stored in the trust server for TF-IDF calculation.
- **BuildIndex**: An index for searching the encrypted data is created by calculating the scores for each unique term $term\_id_{ij}$. Score calculation is a mode of ranking or determining the relevance of each term in the document set. TF-IDF relevance calculation mechanism is used in the proposed scheme. It involves two parameters, the term frequency (TF) and inverse document frequency (IDF). Term frequency represents the number of times a term t occurs in a document D_i. TF value scales up if the term t occurs more times in the document.

$$TF(t,d)=\frac{\text{Count of term t in document d}}{\text{Total number of unique terms in the document d}} \quad (2)$$

IDF represents a term's importance among a set of documents. It scales down the term which occurs in most of the documents.

$$IDF(t)=\log\left(\frac{\text{Total number of documents}}{\text{Number of documents in which the t occurs}}\right) \quad (3)$$

$$tfidf(t,d) = TF(t,d)*IDF(t) \quad (4)$$

The calculated TF-IDF is encrypted to Enc(tfidf) using the simple homomorphic encryption scheme [12] . The key p is chosen such that it is an odd integer, chosen from some interval $p \in [2^{\eta-1}, 2^{\eta})$ where η represents the bit-length of the secret key. The key is assigned to the user by trust server on upload.

$$Enc(tfidf) = pq + 2r + tfidf \quad (5)$$

- **Trapdoor Generation**: When a user initiates a search, all the sharer information is taken. With t keywords of interest as input, the semantics of each term is obtained from the NLP library. The checksums for the corresponding search keyword and any semantics if exists are computed for each document encrypted using the key provided by trust server. The key is accessible to the search user only if the owner has mentioned the particular user as a sharer. Thus the trapdoor generated as $T=\{Enc(CS)_1, Enc(CS)_2... Enc(CS)_{t+s}\}$ where s is the number of keywords obtained as a result of semantic extension
- **Query**: The trapdoor is send to the cloud sever where the corresponding Enc(tfidf) values for the matched Enc(CS) are added to get the actual rank of each document based on the query. The actual score is obtained by decryption [12] of the Enc(tfidf) as given in (6).

$$tfidf= (Enc(tfidf) \bmod p) \bmod 2 \quad (6)$$

B. Supporting Semantic Search

A semantic keyword search can improve search accuracy by understanding intent of the search user and the meanings of the terms. It helps in retrieving the exactly matched files. NLP library is used to find the related words and meanings of each search keyword. Then the search query is reformulated as semantically extended query.

C. Supporting Dynamic Update

It is necessary to facilitate update on the data already stored in the cloud server. Such an update includes insertion of new documents and deletion of the existing ones. The major problem associated with the update is the index update operation. Since the TF-IDF score value is used for ranking, an insertion or deletion will induce a change in the IDF value of all the terms in the existing index. It is required to maintain the metadata for recalculation of IDF i.e. the total number of documents and the number of documents in which each term is present. It was a difficult task to manage the term occurrence completely without a server to get it available for update. The trust server serves the purpose by making the corresponding data available only to the data owner. On an update the TF-IDF is recalculated for all terms, encrypted and updated to the cloud server.

V. PERFORMANCE ANALYSIS

A. *Efficiency*

The proposed scheme reduces the storage overhead of the organization storing millions of documents in the cloud server. The encrypted searchable index also requires a storage space. With the hashed technique of index creation, the overhead of storing dummy keywords is eliminated. Since the hashed checksum is encrypted and stored, no other details about the documents are revealed to the cloud server. Thus the need for dummy keywords are eliminated and there by utilizing only the space for the required unique terms. Thus the index size is reduced in the proposed scheme. The Fig. 3 shows the index size comparison of existing system and the proposed system. The Index Size(Old) and Index Size(New) denotes the size of the index in the existing system and proposed system respectively.

The proposed scheme reduces the computational overhead when compared to the previous MRSE schemes. In the existing MRSE schemes a matrix with a high dimension has to be maintained and the encryption of the index requires multiplication operation which induces much computational overhead. In the proposed system index is constructed for each document unlike the existing system where the index is constructed based on the dictionary vector which contains all the unique terms in the document set. In the MRSE scheme there are entries of zero in the index when a term in the dictionary is not present in that document. In simple terms, the operation $E_i = \{M_1I_i', M_2I_i''\}$ involves multiplication on the terms which are not in the document also in MRSE schemes. In the proposed scheme the encryption is done for individual terms. The index encryption is done with the simple homomorphic encryption scheme [12] which involves addition operation only on the tfidf value.

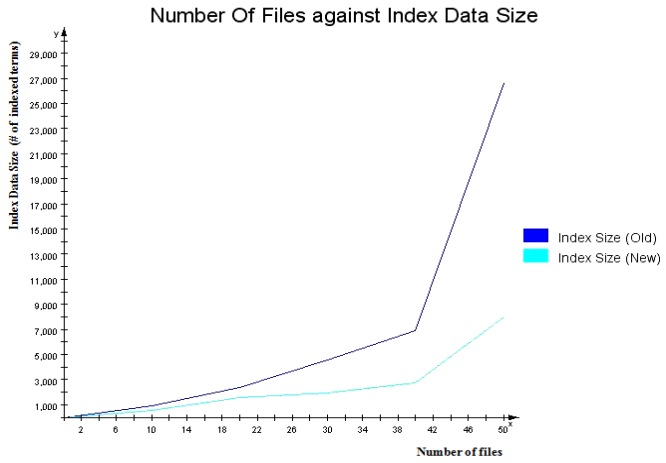

Fig. 3. Index Data Size plotted against the Number of files

The proposed scheme reduces the computational overhead when compared to the previous MRSE schemes. In the existing MRSE schemes a matrix with a high dimension has to be maintained and the encryption of the index requires multiplication operation which induces much computational overhead. In the proposed system index is constructed for each document unlike the existing system where the index is constructed based on the dictionary vector which contains all the unique terms in the document set. In the MRSE scheme there are entries of zero in the index when a term in the dictionary is not present in that document. In simple terms, the operation $E_i = \{M_1I_i', M_2I_i''\}$ involves multiplication on the terms which are not in the document also in MRSE schemes. In the proposed scheme the encryption is done for individual terms. The index encryption is done with the simple homomorphic encryption scheme [12] which involves addition operation only on the tfidf value.

During trapdoor generation, the existing MRSE schemes involves a query vector generation based on the dictionary where non query keywords also involves computation. This computational overhead is eliminated in the proposed system. The hashed method makes it independent of the term-id list for trapdoor generation. The efficient scheme computes the checksum for the corresponding query keywords and encrypt them and send to the cloud server. The score calculation requires only addition operation on all the matched query keywords. Thus the search speed is increased. Fig. 4. shows the search time comparison for the existing and proposed system. Search Time (Old) and Search Time (New) denotes the search times for the existing and proposed systems respectively.

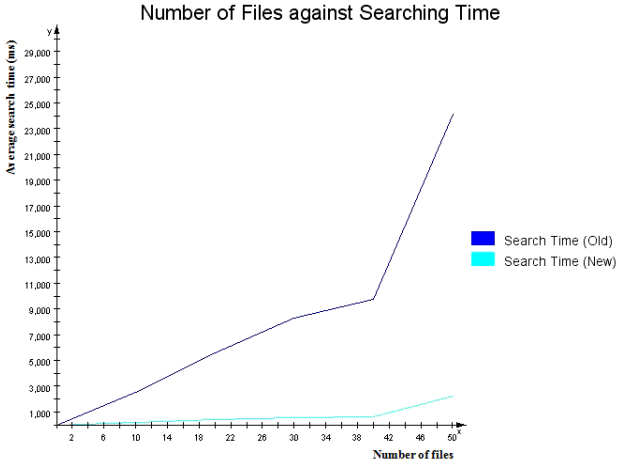

Fig. 4. Searching Time plotted against the Number of files

VI. CONCLUSION

In this paper, an efficient scheme for the semantic search in the encrypted cloud data is proposed. A hashed index is constructed and encrypted with a secure homomorphic encryption which is sufficient for the ranked search. Ranking is done based on TF-IDF. The semantics for each query keywords are determined by utilizing the NLP library available and thereby creating an extended query. The proposed scheme concentrate on improving the search efficiency by reducing the storage and computational overheads compared to previous systems. Future works in the privacy preserving side will enhance the efficiency of the proposed system.

ACKNOWLEDGMENT

The authors wish thanks to the Management and Principal and Head Of the Department (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] E.-J. Goh, “Secure Indexes,” Cryptology ePrint Archive, http://eprint.iacr.org/2003/216.

[3] D. Boneh, G.D. Crescenzo, R. Ostrovsky, and G. Persiano, PublicKey Encryption with Keyword Search,” Proc. Int’l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT), 2004.

[4] Y.-C. Chang and M. Mitzenmacher, “Privacy Preserving Keyword Searches on Remote Encrypted Data,” Proc. Third Int’l Conf. Applied Cryptography and Network Security, 2005..

[5] P. Golle, J. Staddon, and B. Waters, “Secure Conjunctive Keyword Search over Encrypted Data,” Proc. Applied Cryptography and Network Security, pp. 31-45, 2004.

[6] L. Ballard, S. Kamara, and F. Monrose, “Achieving Efficient Conjunctive Keyword Searches over Encrypted Data,” Proc. Seventh Int’l Conf. Information and Comm. Security (ICICS ’05), 2005.

[7] E. Shen, E. Shi, and B. Waters, “Predicate Privacy in Encryption Systems,” Proc. Sixth Theory of Cryptography Conf. Theory of Cryptography (TCC), 2009.

[8] W.K. Wong, D.W. Cheung, B. Kao, and N. Mamoulis, “Secure kNN Computation on Encrypted Databases,” Proc. 35th ACM SIGMOD Int’l Conf. Management of Data (SIGMOD), pp. 139-152, 2009.

[9] Ning Cao, Cong Wang, Member, Ming Li, Kui Ren, Senior Member,and Wenjing Lou,"Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data" Proc. IEEE Transactions On Parallel And Distributed Systems, Vol. 25, No. 1, 2014.

[10] I.H. Witten, A. Moffat, and T.C. Bell, Managing Gigabytes:Compressing and Indexing Documents and Images. organ Kaufmann Publishing, May 1999.

[11] Xingming Sun, Yanling Zhu, Zhihua Xia and Lihong Chen,"Privacy-Preserving Keyword based Semantic Search over Encrypted Cloud Data" Proc. International Journal of Security and Its Applications Vol.8, No.3 (2014), pp.9-20

[12] Marten van Dijk, Craig Gentry, Shai Halevi, Vinod Vaikuntanathan, " Fully Homomorphic Encryption over the Integers", 2010.