International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 2, Issue 4, April 2015

Checksum Validated Individual Page Updates on Encrypted Keyword

[1]L.Harikrishna,[2]M.Harrish,[3]R.Mahendar,[4]V.Mohammed Rameez Khan,[5]T.Veeramani
[1][2][3][4]Final Year, Department of Information Technology, Valliammai Engineering College , Chennai-603203, Tamil nadu, India.
[5]Assistant Professor , Department of Information Technology, Valliammai Engineering College , Chennai-603203, Tamil nadu, India.
[1]harikrishnaloganathan12@gmail.com,[2]hrrsh.mj@gmail.com,[3]mahendar.ravichandran@gmail.com,[4]mh.khan92@gmail.com,[5]veeramaniphd@gmail.com

Abstract- **Information sharing is the key goal of Cloud Storage servers. It allows storage of sensitive and large volume of data with limited cost and high access benefits. Security must be given due importance for the cloud data with utmost care to the data and confidence to the data owner. But this limits the utilization of data through plain text search. Hence an excellent methodology is required to match the keywords with encrypted cloud data. The proposed approach similarity measure of "coordinate matching" combined with "inner product similarity" quantitatively evaluates and matches all relevant data with search keyword to arrive at best results .In this approach, each document is associated with a binary vector to represent a keyword contained in the document. The search keyword is also described as a binary vector, so the similarity could be exactly measured by the inner product of the query vector with the data vector. The inner product computation and the two multi-keyword ranked search over encrypted data (MRSE) schemes ensures data privacy and provides detailed information about the dynamic operation on the data set and index. To further refine these search results, "stop word" and "stemming" techniques have been used. Also, a checksum value for each page is stored using checksum validation algorithm and individual pages are updated in the cloud which reduces the usage cost and hence improves the experience of the user.**

Keywords: **Cloud computing, searchable encryption, privacy-preserving, keyword search, ranked search, checksum validation**

I. INTRODUCTION

CLOUD computing is the long dreamed vision of computing as a utility, where cloud customers can remotely store their data into the cloud so as to enjoy the on-demand high-quality applications and services from a shared pool of configurable computing resources. To protect data privacy and combat unsolicited accesses in the cloud and beyond, sensitive data, for example, e-mails, personal health records and so on, may have to be encrypted by data owners before outsourcing to the commercial public cloud. However this obsoletes the traditional data utilization service based on plaintext keyword search. The trivial solution of downloading all the data and decrypting locally is clearly impractical, due to the huge amount of bandwidth cost in cloud scale systems. To meet the effective data retrieval need, the large amount of documents demand the cloud server to perform result relevance ranking, instead of returning undifferentiated results. Such ranked search system enables data users to find the most relevant information quickly, rather than burdensomely sorting through every match in the content collection.

Ranked search can also elegantly eliminate unnecessary network traffic by sending back only the most relevant data, which is highly desirable in the "pay-as-you-use" cloud paradigm. For privacy protection, such ranking operation, however, should not leak any keyword related information. On the other hand, to improve the search result accuracy as well as to enhance the user searching experience, it is also necessary for such ranking system to support multiple keywords search, as single keyword search often yields far too coarse results. Each keyword in the search request is able to help narrow down the search result further. Among various multi-keyword semantics, we choose the efficient similarity measure of "coordinate matching", i.e., as many matches as possible, to capture the relevance of data documents to the search query. Specifically, we use "inner product similarity", i.e., the number of query keywords appearing in a document, to quantitatively evaluate such similarity measure of that document to the search query.
Fig.

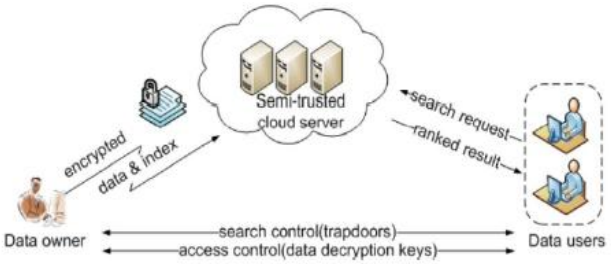

Fig.1 Architecture of search over encrypted cloud data

This technique treats encrypted data as documents and allows a user to securely search through a single keyword and retrieve documents of interest. Although some recent designs have been proposed to support multi-keyword ranked search over encrypted cloud data (MRSE) search attempt to enrich the search flexibility, they are still not adequate to provide users with acceptable result ranking functionality. In this paper, for the first time, we define and solve this problem by encrypting using Triple DES algorithm, while preserving strict system wise privacy in the cloud computing paradigm. During the index construction, each document is associated with a binary vector as a sub-index where each bit represents whether corresponding keyword is contained in the document. The search query is also described as a binary vector where each bit means whether corresponding keyword appears in this search request, so the similarity could be exactly measured by the inner product of the query vector with the data vector. However, directly outsourcing the data vector or the query vector will violate the index privacy or the search privacy. To meet the challenge of supporting such multi-keyword semantic without privacy breaches, we propose a basic idea for the MRSE using secure inner product computation, which is adapted from a secure k-nearest neighbour (kNN) technique.

Also to reduce the cost of cloud usage, only individual pages are updated using checksum validation technique. According to this, the checksum value of each page is stored in a table. On editing a specific page, it is identified using this checksum value and those pages are updated in the cloud.

II. PROBLEM FORMULATION

2.1 System Model

Considering a cloud data hosting service involving three different entities, as illustrated in Fig. 1: the data owner, the data user, and the cloud server. The data owner has a collection of data documents F to be outsourced to the cloud server in the encrypted form C. To enable the searching capability over C for effective data utilization, the data owner, before outsourcing, will first build an encrypted searchable index I from F, and then outsource both the index I and the encrypted document collection C to the cloud server. To search the document collection for t given keywords, an authorized user acquires a corresponding trapdoor T through search control mechanisms, for example, broadcast encryption[1]. Upon receiving T from a data user, the cloud server is responsible to search the index I and return the corresponding set of encrypted documents. To improve the document retrieval accuracy, the search result should be ranked by the cloud server according to some ranking criteria (e.g., coordinate matching, as will be introduced shortly). Moreover, to reduce the communication cost, the data user may send an optional number k along with the trapdoor T so that the cloud server only sends back top-k documents that are most relevant to the search query. Finally, the access control mechanism [2] is employed to manage decryption capabilities given to users and the data collection can be updated in terms of inserting new documents, updating existing documents, and deleting existing documents.

2.2 Threat Model

The cloud server is considered as "honest-but-curious" in our model, which is consistent with related works on cloud security [2], [3]. Specifically, the cloud server acts in an "honest" fashion and correctly follows the designated protocol specification.However, it is "curious" to infer and analyze data (including index) in its storage and message flows received during the protocol so as to learn additional information. Based on what information the cloud server knows, we consider two threat models with different attack capabilities as follows.Known ciphertext model. In this model, the cloud server is supposed to only know encrypted data set C and searchable index I, both of which are outsourced from the data owner.Known background model. In this stronger model, the cloud server is supposed to possess more knowledge than what can be accessed in the known ciphertext model. Such information may include the correlation relationship of given searchrequests (trapdoors), as well as the data set related statistical information. As an instance of possible attacks in this case,the cloud server could use the known trapdoor information combined with document/keyword frequency [4] to deduce/identify certain keywords in the query.

2.3 Design Goals

To enable ranked search for effective utilization of outsourced cloud data under the aforementioned model, our system design should simultaneously achieve security and performance guarantees as follows.

Multi-keyword ranked search. To design search schemes which allow multi-keyword query and provide result similarity ranking for effective data retrieval, instead of returning undifferentiated results.

Privacy-preserving. To prevent the cloud server from learning additional information from the data set and the

index, and to meet privacy requirements specified in Section 3.2.

Checksum validation. To minimize the cloud usage cost for the users and also to boost the user experience, only individual pages needs to be updated using the checksum values.

Efficiency. Above goals on functionality and privacy should be achieved with low communication and computation overhead.

2.4 Notations

F—the plaintext document collection, denoted as a set of m data documents F =(F1,F2, . . . Fm).

C—the encrypted document collection stored in the cloud server, denoted as C = (C1,C2, . . . Cm).

W—the dictionary, i.e., the keyword set consisting of n keyword, denoted as W =(W1,W2, . . .,Wn).

I—the searchable index associated with C, denoted as (I1 , I2 , . . . , Im) where each subindex Ii is built for Fi.

Wj—the subset of W, representing the keywords in search request, denoted as W =(Wj1,Wj2 , . . Wjt).

TW—the trapdoor for the search request W.

FW —the ranked id list of all documents according to their relevance to W.

2.5 Preliminary on Coordinate Matching

As a hybrid of conjunctive search and disjunctive search,"coordinate matching" [5] is an intermediate similarity measure which uses the number of query keywords appearing in the document to quantify the relevance of that document to the query. When users know the exact subset of the data set to be retrieved, Boolean queries perform well with the precise search requirement specified by the user. In cloud computing, however, this is not the practical case, given the huge amount of outsourced data.

Therefore, it is more flexible for users to specify a list of keywords indicating their interest and retrieve the most relevant documents with a rank order.

III. FRAMEWORK AND PRIVACY REQUIREMENTS FOR MRSE

In this section, we define the framework of multi-keyword ranked search over encrypted cloud data (MRSE) and establish various strict systemwise privacy requirements for such a secure cloud data utilization system.

3.1 MRSE Framework

For easy presentation, operations on the data documents are not shown in the framework since the data owner could easily employ the traditional symmetric key cryptography to encrypt and then outsource data. With focus on the index and query, the MRSE system consists of four algorithms as follows:

Setup (1l)Taking a security parameter l as input, the data owner outputs a symmetric key as SK.

BuildIndex (F,SK). Based on the data set F, the data owner builds a searchable index I which is encrypted by the symmetric key SK and then outsourced to the cloud server. After the index construction, the document collection can be independently encrypted and outsourced.

Trapdoor (W). With t keywords of interest in W as input, this algorithm generates a corresponding trapdoor TW

Query (Tw,k,I). When the cloud server receives a query request as (Tw, k), it performs the ranked search on the index I with the help of trapdoor Tw,and finally returns Fw, the ranked id list of top-k documents sorted by their similarity with W.

Neither the search control nor the access control is within the scope of this paper. While the former is to regulate how authorized users acquire trapdoors, the later is to manage users' access to outsourced documents.

3.2 Privacy Requirements for MRSE

The representative privacy guarantee in the related literature,such as searchable encryption, is that the server should learn nothing but search results. With this general privacy description, we explore and establish a set of strict privacy requirements specifically for the MRSE framework.

As for the *data privacy*, the data owner can resort to the traditional symmetric key cryptography to encrypt the data before outsourcing, and successfully prevent the cloud server from prying into the outsourced data. With respect to the index privacy, if the cloud server deduces any association between keywords and encrypted documents from index, it may learn the major subject of a document, even the content of a short document [4]. Therefore, the searchable index should be constructed to prevent the cloud server from performing such kind of association attack. While data and index privacy guarantees are demanded by default in the related literature,various search privacy requirements involved in the query procedure are more complex and difficult to tackle as follows

Keyword privacy. As users usually prefer to keep their search from being exposed to others like the cloud server,the most important concern is to hide what they are searching, i.e., the keywords indicated by the corresponding trapdoor. Although the trapdoor can be generated in a cryptographic way to protect the query keywords, the cloud server could

do some statistical analysis over the search result to make an estimate.As a kind of statistical information,document frequency (i.e.,the number of documents containing the keyword) is sufficient to identify the keyword with high probability [6]. When the cloud server knows some background information of the data set, this keyword specific information may be utilized to reverseengineer the keyword.

Trapdoor unlinkability. The trapdoor generation function should be a randomized one instead of being deterministic. In particular, the cloud server should not be able to deduce the relationship of any given trapdoors, for example, to determine whether the two trapdoors are formed by the same search request. Otherwise, the deterministic trapdoor generation would give the cloud server advantage to accumulate frequencies of different search requests regarding different keyword(s), which may further violate the aforementioned keyword privacy requirement. So the fundamental protection for trapdoor unlinkability is to introduce sufficient nondeterminacy into the trapdoor generation procedure.

Access pattern. Within the ranked search, the access pattern is the sequence of search results where every search result is a set of documents with rank order. Specifically, the search result for the query keyword set fW is denoted as FW. consisting of the id list of all documents ranked by their relevance to fW. Then the access pattern is denoted as (F~W1,F~W2,. .)which are the results of sequential searches.Although a few searchable encryption works, for example,[7]has been proposed to utilize private information retrieval (PIR) technique [8], to hide the access pattern, our proposed schemes are not designed to protect the access pattern for the efficiency concerns.This is because any PIRbased technique must "touch" the whole data set outsourced on the server which is inefficient in the large-scale cloud system.

IV. PRIVACY-PRESERVING AND EFFICIENT MRSE

To efficiently achieve multi-keyword ranked search, we propose to employ "inner product similarity" [5] to quantitatively evaluate the efficient similarity measure "coordinate matching."Specifically,Di is a binary data vector for document Fi where each bit Di[j] € {0,1} represents the existence of the corresponding keyword Wj in that document, and Q is a binary query vector indicating the keywords of interest where each bit Q[j] € {0,1} represents the existence of the corresponding keyword Wj in the query fW. The similarity score of document Fi to query w is therefore expressed as the inner product of their binary column vectors,i.e.,Di . Q. For the purpose of ranking, the cloud server must be given the capability to compare the similarity of different documents to the query. But, to preserve strict systemwise privacy, data vector Di, query vector Q and their inner product Di . Q should not be exposed to the cloud server. In this section,we first propose a basic idea for the MRSE using secure inner product computation, which is adapted from a secure kNN technique, and then show how to significantly improve it to be privacy-preserving against different threat models in the MRSE framework in a step-by-step manner.We further discuss supporting more search semantics and dynamic operation.

4.1 Secure Inner Product Computation

In the secure kNN scheme [9], euclidean distance between a data record pi and a query vector q is used to select k nearest database records. The secret key is composed of one (d + 1)-bit vector as S and two (d + 1) x (d + 1) invertible matrices as {M1,M2},where d is the number of fields for each record pi. First, every data vector pi and query vector q are extended to (d + 1)-dimension vectors as pi and q, where the (d + 1)th dimension is set to -0.5||pi2|| and 1, respectively. Besides, the query vector ~q is scaled by a random number r > 0 as (rq,r). Then, pi is split into two random vectors as {pi,pj}, and qn is also split into two random vectors as {q1,qn}. Note here that vector S functions as a splitting indicator. Namely, if the jth bit of S is 0, p1[j] and pn[j] are set as the same as pi[j] and q1[j] are set to two random numbers so that their sum is equal to q[j]; if the *j*th bit of S is 1, the splitting process is similar except that pi and q are switched.

As the MRSE is using the inner product similarity instead of the euclidean distance, we need to do some modifications on the data structure to fit the MRSE framework. One way to do that is by eliminating the dimension extension, the final result changes to be the inner product as rpi . q. While the encryption of either data record or query vector involves two multiplications of a d x d matrix and a d-dimension vector with complexity O(d2),the final inner product computation involves two multiplications of two d-dimension vectors with complexity O(d). In the known ciphertext model, the splitting vector S is unknown,so p1[j] and pn[j] are considered as two random d-dimensional vectors. To solve the linear equations created by the encryption of data vectors, we have 2dm unknowns in m data vectors and 2d2 unknowns in {M1,M2}.Since we have only 2dm equations, which are less than the number of unknowns, there is no sufficient information to solve either data vectors or {M1,M2}. Similarly, q1[j] and qn[j] are also considered as two random d-dimensional vectors. To solve the linear equations created by the encryption of query vectors, we have 2d unknowns in two query vectors and 2d2 unknowns in {M1,M2}. Since we have only 2d equations here, which are less than the number of unknowns, there is no sufficient information to solve either query vectors or {M1,M2}. Hence, we believe

that without prior knowledge of secretkey, neither data vector nor query vector, after such a series of processes like splitting and multiplication, can berecovered by analyzing their corresponding ciphertexts.

4.2 Analysis

We analyze this MRSE scheme from three aspects of design goals described in Section 2.

ible tradeoff parameter among search accuracy and security. From the consideration of effectiveness, ∂ is expected to be smaller so as to obtain high precision indicating the good purity of retrieved documents. To quantitatively evaluate the search accuracy, we set a measure as precision pk to capture the fraction of returned top-k documents that are included in the real topk list.

As for the efficiency, our inner product-based MRSE scheme is an outstanding approach from the performance perspective. In the steps like BuildIndex or Trapdoor, the generation procedure of each subindex or trapdoor involves two multiplications of a (n+2) x (n+2) matrix and a (n+2)-dimension vector. In the Query, the final similarity score is computed through two multiplications of two (n+2)-dimension vectors.

Privacy. As for the data privacy, traditional symmetric key encryption techniques could be properly utilized here and is not within the scope of this paper. The index privacy is well protected if the secret key SK is kept confidential since such vector encryption method has been proved to be secure in the known ciphertext model [7]. Although we add two more dimensions to the vectors compared to the adapted secure inner product computation, the number of equations as 2(n+2)m is still less than the number of unknowns as the sum of 2(n+2)m unknowns in m data vectors and 2d2 unknowns in {M1;M2}. With the randomness introduced by the splitting process and the random numbers r, and t, our basic scheme can generate two totally different trapdoors for the same query f W. This nondeterministic trapdoor generation can guarantee the trapdoor unlinkability which is an unsolved privacy leakage

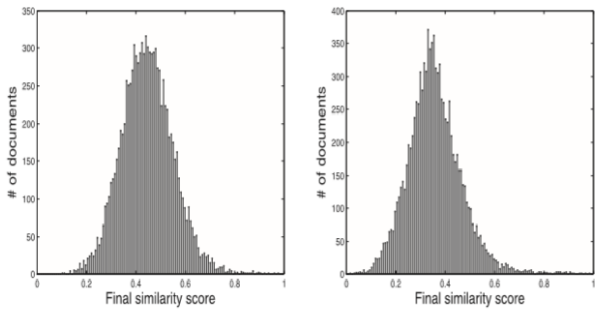

Fig. 2. Distribution of final similarity score with different standard deviations, 10k documents, 10 query keywords. (a) $\sigma = 1$. (b) $\sigma = 0.5$.

problem in related symmetric key-based searchable encryption schemes because of the deterministic property of trapdoor generation [1]. Moreover, with properly selected parameter ∂ for the random factor "i, even the final score results can be obfuscated very well, preventing the cloud server from learning the relationships of given trapdoors and the corresponding keywords. Note that although ∂ is expected to be small from the effectiveness point of view, the small one will introduce small obfuscation into the final similarity scores, which may weaken the protection of keyword privacy and trapdoor unlinkability. As shown in Fig. 2, the distribution of the final similarity scores with smaller ∂ will enable the cloud server to learn more statistical information about the original similarity scores, and therefore ∂ should be set large enough from the consideration of privacy.

4.3 MRSE_II: Privacy-Preserving Scheme in Known Background Model When the cloud server has knowledge of some background information on the outsourced data set, for example, the correlation relationship of two given trapdoors, certain keyword privacy may not be guaranteed anymore by the MRSE_I scheme. This is possible in the known background model because the cloud server can use scale analysis as follows to deduce the keyword specific information, for example, document frequency, which can be further combined with background information to identify the keyword in a query at high probability. After presenting how the cloud server uses scale analysis attack

TABLE 1
K_3 Appears in Every Document

Doc	Query for $\{K_1, K_2, K_3\}$	Query for $\{K_1, K_2\}$
1	$x_1 = 3, y_1 = r(3+\varepsilon_1)+t$	$x'_1 = 2, y'_1 = r'(2+\varepsilon_1)+t'$
2	$x_2 = 2, y_2 = r(2+\varepsilon_2)+t$	$x'_2 = 1, y'_2 = r'(1+\varepsilon_2)+t'$
3	$x_3 = 1, y_3 = r(1+\varepsilon_3)+t$	$x'_3 = 0, y'_3 = r'(0+\varepsilon_3)+t'$

to break the keyword privacy, we propose a more advanced MRSE scheme to be privacy-preserving in the known background model

V. PERFORMANCE ANALYSIS

In this section, the whole experiment system is implemented by C language on a Linux Server with Intel Xeon Processor 2.93 GHz. The performance of our technique is evaluated regarding the efficiency of four proposed MRSE schemes, as well as the tradeoff between search precision and privacy.

5.1 Precision and Privacy

As presented in Section 4, dummy keywords are inserted into each data vector and some of them are selected in every query. Therefore, similarity scores of documents will be not exactly accurate.

Fig. 3a shows that the precision in MRSE scheme is evidently affected by the standard deviation of the random variable ". From the consideration of effectiveness,

standard deviation is expected to be smaller so as to obtain high precision indicating the good purity of retrieved documents. However, user's rank privacy may have been

partially leaked to the cloud server as a consequence of small .

5.2 Efficiency

5.2.1 Index Construction

To build a searchable subindex Ii for each document Fi

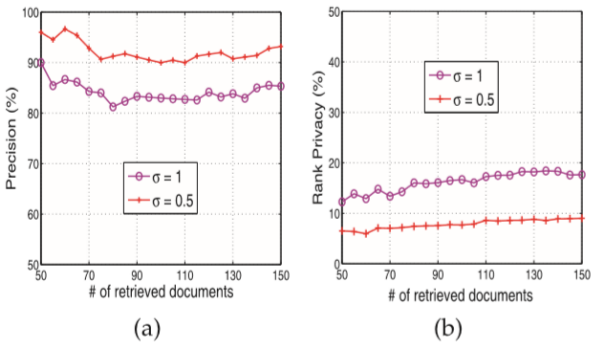

Fig. 3. With different choice of standard deviation σ for the random variable ε, there exists tradeoff between (a) Precision, and (b) Rank Privacy.

standard deviation is expected to be smaller so as to obtain high precision indicating the good purity of retrieved documents. However, user's rank privacy may have been partially leaked to the cloud server as a consequence of small .

5.2 Efficiency

5.2.1 Index Construction

To build a searchable subindex Ii for each document Fi in the data set F, the first step is to map the keyword set extracted from the document Fi to a data vector Di, followed by encrypting every data vector. Fig. 4a shows that, the time cost of building the whole index is nearly linear with the size of data set since the time cost of building each subindex is fixed. Fig. 4b shows that the number of keywords indexed in the dictionary determines the time cost of building a subindex. Figs. 4a and 4b. Both the MRSE_I_TF and the MRSE_II_TF, presented in Sections

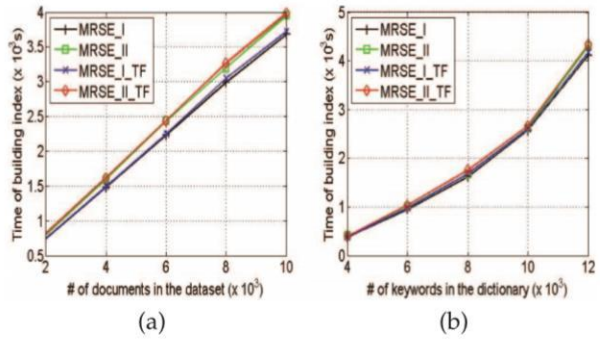

Fig. 4. Time cost of building index. (a) For the different size of data set with the same dictionary, $n = 4{,}000$. (b) For the same data set with different size of dictionary, $m = 1{,}000$.

4.4 and 4.5, respectively, introduce more computaion during the index construction since we need to collect the term frequency information for each keyword in every document and then perform the normalization calculation. But, as shown in both figures, such additional computation in the TF x IDF weighting rule is insignificant considering much more computation are caused by the splitting process and matrix multiplication. Although the time of building index is not a negligible overhead for the data owner, this is a one-time operation before data outsourcing. The size of subindex is absolutely linear with the dimensionality of data vector which is determined by the number of keywords in the dictionary. The sizes of subindex are very close in the two MRSE schemes because of trivial differences in the dimensionality of data vector.

5.2.2 Trapdoor Generation

The time to generate a trapdoor is greatly affected by the number of keywords in the dictionary. Like index construction, every trapdoor generation incurs two multiplications of a matrix and a split query vector, where the dimensionality of matrix or query vector is different in two proposed schemes and becomes larger with the increasing size of dictionary. Like the subindex generation, the difference of costs to generate trapdoors is mainly caused by the different dimensionality of vector and matrices in the two MRSE schemes. More importantly, it shows that the number of query keywords has little influence on the overhead of trapdoor generation, which is a significant advantage over related works on multi-keyword searchable encryption.

5.2.3 **Query**

Query execution in the cloud server consists of computing and ranking similarity scores for all documents in the data set. The query time is dominated by the number of documents in the data set while the number of keywords in the query has very slight impact on it like the cost of trapdoor generation above.With respect to the communication cost in Query, the size of the trapdoor is the same as that of the subindex , which keeps constant given the same dictionary, no matter how many keywords are contained in a query. While the computation and communication cost in the query procedure is linear with the number of query keywords in other multiple-keyword search schemes [6], [8], our proposed schemes introduce nearly constant overhead while increasing the number of query keywords. Therefore, our schemes cannot be compromised by timing-based side channel attacks that try to differentiate certain queries based on their query time

RELATED WORKS:

MRSE:

For easy presentation, operations on the data documents are not shown in the framework since the data owner could easily employ the traditional symmetric key cryptography to encrypt and then outsource data. With focus on the index

and query, the MRSE system consists of four algorithms as follows:

✓ **Setup (1ı) :** Taking a security parameter **l** as input, the data owner outputs a symmetric key as **SK**.

✓ **Build-Index (F,SK) :** Based on the data set **F**, the data owner builds a searchable index **I** which is encrypted by the symmetric key **SK** and then outsourced to the cloud server. After the index construction, the document collection can be independently encrypted and outsourced.

✓ **Trapdoor (W) :** With **t** keywords of interest in **W** as input, this algorithm generates a corresponding trapdoor **Tw**.

Query (Tw,k,I) : When the cloud server receives a query request as **(Tw , k)**, it performs the ranked search on the index I with the help of trapdoor **Tw**, and finally returns **Fw** , the ranked id list of top-k documents sorted by their similarity with **W**.

CHECKSUM VALIDATION:

A checksum is a count of the number of bits in a transmission unit that is included with the unit so that the receiver can check to see whether the same number of bits arrived. If the counts match, it's assumed that the complete transmission was received. Both **TCP** and **UDP** communication layers provide a checksum count and verification as one of their services.

STOP WORDS:

Stop words are words which are filtered out before or after processing of natural language data (text). There is no single universal list of stop words used by all processing of natural language tools, and indeed not all tools even use such a list. Some tools specifically avoid removing these stop words to support phrase search. Any group of words can be chosen as the stop words for a given purpose. For some search engines, these are some of the most common, short function words, such as **the**, **is**, **at**, **which** and **on**. In this case, stop words can cause problems when searching for phrases that include them, particularly in names such as 'The Who', 'The The', or 'Take That'. Other search engines remove some of the most common words—including lexical words, such as "**want**"—from a query in order to improve performance.

STEMMING:

A stemming algorithm is a process of linguistic normalisation, in which the variant forms of a word are reduced to a common form, for example,

connection
connections
connective ---> connect
connected
connecting

It is important to appreciate that we use stemming with the intention of improving the performance of IR systems. It is not an exercise in etymology or grammar. In fact from an etymological or grammatical viewpoint, a stemming algorithm is liable to make many mistakes. In addition, stemming algorithms - at least the ones presented here - are applicable to the written, not the spoken, form of the language.For some of the world's languages, Chinese for example, the concept of stemming is not applicable, but it is certainly meaningful for the many languages of the Indo-European group. In these languages words tend to be constant at the front, and to vary at the end:

-ion
-ions
connect -ive
-ed
-ing

The variable part is the `ending', or `suffix'. Taking these endings off is called `suffix stripping' or `stemming', and the residual part is called the stem.

CONCLUSION:

In this paper,we define and solve the problem of MRSE scheme and provide a privacy for search using multiple keywords. Among various multi-keyword semantics, we choose the efficient similarity measure of "**coordinate matching**", i.e., as many matches as possible, to capture the relevance of data documents to the search query. Specifically, we use "**inner product similarity**", i.e., the number of query keywords appearing in a document, to quantitatively evaluate such similarity measure of that document to the search query.Also checksum validation technique is used.According to this technique a checksum value for each page is stored and individual pages are updated in the cloud which reduces the usage cost and hence improves the experience of the user.

In our future work, we will explore deep content search using inner product similarity to become a reality.

REFERENCES:

[1] R. Curtmola, J.A. Garay, S. Kamara, and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions,"Proc. 13th ACM Conf. Computer and Comm. Security (CCS '06), 2006.

[2] S.Yu, C.Wang, K.Ren, and W.Lou, "Achieving Secure, Scalable,and Fine-Grained Data Access Control in Cloud Computing,"
Proc. IEEE INFOCOM, 2010.

[3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing,"
Proc. IEEE INFOCOM, 2010.

[4]S. Zerr, E. Demidova, D. Olmedilla, W. Nejdl, M. Winslett, and S. Mitra, "Zerber: r-Confidential Indexing for Distributed Documents," Proc. 11th Int'l Conf. Extending Database Technology (EDBT '08), pp. 287-298, 2008.

[5] I.H. Witten, A. Moffat, and T.C. Bell, Managing Gigabytes:Compressing and Indexing Documents and Images. Morgan Kaufmann Publishing, May 1999.
[6] 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 (EDBT '09), pp. 439-449, 2009.
[7] R. Brinkman, "Searching in Encrypted Data," PhD thesis, Univ. of Twente, 2007.
[8] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Cryptography from Anonymity," Proc. IEEE 47th Ann. Symp. Foundationsof CS,pp.239-248, 2006.
[9] 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.