An Efficient Multi-keyword Ranked Retrieval Scheme with Johnson-Lindenstrauss Transform Over Encrypted Cloud Data

Ke Li, Weiming Zhang, Ke Tian, Rundong Liu and Nenghai Yu

***Abstract*—Cloud computing is becoming more and more popular recently, and this trend encourages large corporations and companies to outsource their data to a remote cloud server. However, the concern of data privacy and security is a huge problem which impedes the development of cloud computing. It is essential and necessary to propose a scheme to realize the information retrieval and protect the data privacy at the same time. Since ranked multi-keyword search is a pretty practical question in this area and it has a significant influence on the user experience, we would mainly focus on the scheme to achieve high performance and low cost multi-keyword ranked search. In this paper, we proposed a novel search scheme with Johnson-Lindenstrauss transform (J-L scheme for short) and a technique called "optimised maximum query" to live up to our efficiency demand. Furthermore, we established a complete system to guarantee a set of strict privacy requirements. Extensive experiments on real-world database are presented to show our proposed scheme's efficiency and accuracy.**

***Keywords*—multi-keyword ranked search, privacy preserving, cloud computing**

I. Introduction

Cloud computing is prevalent in our daily life everywhere. The long-held dream of computing as a utility, has a significant influence on the IT industry and makes the software more attractive and shapes the way of everyone's life [1]. In fact, cloud computing is defined as a colloquial expression which is used to describe a variety of different computing concepts which involve plenty of computers that are connected through a real-time communication network. Moreover, companies and corporations with large computing tasks are motivated to outsource the difficult tasks to cloud server to make the best of cloud's high performance computing capacity. These data owners would also like to store data into cloud server for convenience and low storage cost. In a word, we can anticipate the huge potential of the cloud computing development.

Even though cloud computing grows rapidly, there are still large number of people afraid of using cloud computing anyway. They mainly concern about data privacy in the cloud computing especially data abuse and information leakage [2]. Internet companies (such as Amazon, Google, Dropbox) have offered low price cloud service to attract customers. But considering that these companies may take advantage of personal data, such as personal health record, for commercial profit probably, we could not completely trust the cloud server. In order to protect data privacy, we need to encrypt the data before outsourcing them to a commercial cloud server. After encrypting the data, we will confront a huge challenge about how to analyze this data , especially how to search the interesting keywords. Information Retrieval (IR) is always the most prominent problem and basic demand for mass data use. However, traditional symmetric encryption demand users to download all the data, decrypt all, and then search keywords like plaintext retrieval. This method not only costs user plenty of time but also is rather impractical. Thus achieving privacy preserving and effective keyword search in the encrypted cloud database is rather important.

In order to meet efficiency demand of information retrieval, Search Encryption (SE) [3] has been proposed to solve this problem. However, directly deploying it to large scale data will cause space and time disadvantages. Therefore, building a secure and efficient search index [4] [5] is necessary. In this paper we mostly concentrate on multi-keyword ranked search because when user executes the information retrieval, he/she will only wants the most related documents. Therefore, how to make an efficient index and sort the search results is significant prominent in our work. Former works have adopt different techniques [6]-[10]. D.Song *et al*. [11] have firstly proposed a scheme only supporting single Boolean keyword search, which is regarded as "coordinate matching". C.Wang *et al* [6] introduced order preserving encryption(OPE) to protect the keyword frequency and realized a single keyword search scheme. Based on the OPE, J. Xu *et al* [10] proposed Two-Step-Ranking to fulfill multi-keyword ranked search. Even though OPE have protected keyword frequency, it will still leak the relationship of frequency comparison. Furthermore, N. Cao *et al* [7] proposed multi-keyword ranked search scheme called MRSE based on secure KNN computation. In MRSE scheme, N. Cao adopted the space vector model to build the index, which means that all of the keywords are defined in a prebuilt dictionary and every keyword is identified by its location in the dictionary. Each article is corresponding to a vector and each position in the vector represents whether or not the corresponding keyword is in this article. Then symmetric encryption (like DES [12], AES [13]) is used for encrypting the documents and two random inverted matrixes are used for encrypting the index. Furthermore, MRSE applies an internal ranking algorithm by using the inner product of two query vectors to determine the most related top-k results. However, this approach has several drawbacks. The most important one is the space consume, the total space consume will increase extremely rapidly when the number of keywords in the dictionary increases. Secondly, MRSE does

978-1-4799-2830-9/14 $31.00 © 2014 IEEE
DOI 10.1109/CLOUDCOM-ASIA.2013.37

not take keyword frequency into consideration, so the files which contain heavy frequencies might not be included in top returned results due to the MRSE method. Thirdly, MRSE doesn't support index update efficiently. For each new keyword to be added to the dictionary, the whole index has to be rebuilt. Z. Xu *et al* [8] proposed MKQE based on the MRSE to improve the efficiency for updating function, but MKQE also did not solve the space consume problem very well.

In this paper, we applied Johnson-Lindenstrauss (JL) Transform [15] to the multi-keyword ranked search. Since this field used to be dominated by the OPE method and secure-KNN method, we are pretty motivated to introduce JL transform technique and present its potential application in this field.

Directly applying JL transform cannot get high efficiency as we need, so we adjust this technique and improve its algorithm to match the efficiency and security demand. To our best knowledge, it is the first time that JL transform is introduced to provide a solution for the multi-keyword ranked search. Then we propose "Optimized Maximum Query" method to make an efficient trapdoor and conquer the problem of low accuracy by directly using JL transform to search. The experiment result on real-world data set shows that our scheme not only guarantees the efficiency, but also reduces the space complexity significantly. Our scheme is flexible and the parameters can be selected with different demand of efficiency and accuracy.

In summary, our contribution can be listed as follows:

- We proposed an efficient scheme based on JL transform to solve the multi-keyword ranked search problem for the first time.
- Our scheme significantly reduces the space complexity and guarantees the efficiency according to the experiments.
- The paraments in our scheme is flexible and we show that potential of application of JL transform in research and practical areas is unlimited.

The rest of our paper is organized as follows: we will describe the basic system model and privacy requirements in Section II. In section III, we will elaborate the JL transform based scheme (J-L scheme) with some analysis. We present the experiment results in Section IV and draw a conclusion in Section V.

II. Problem Formulation

In this Section, we will concentrate on defining the problem. We will describe our searching system model and the privacy requirements based on the analysis. Then we will describe some preliminaries related to our work.

A. System model

Different from traditional plaintext information retrieval, the encrypted database retrieval in cloud computing always have three entities:data owner, remote cloud server and users. The data owner can be divided into individuals or corporations who own a collection of n plain document files

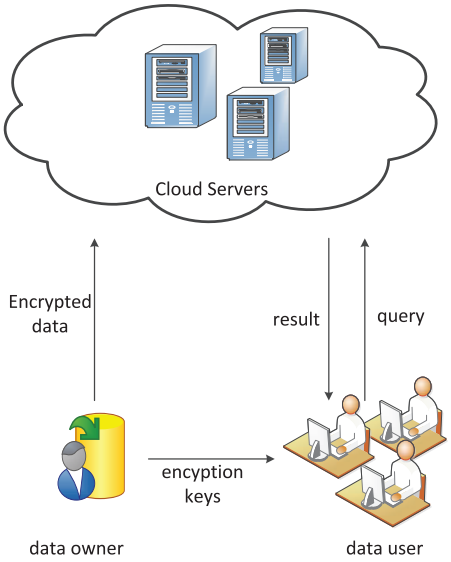

Fig. 1. System Framework of multi-keyword ranked search

$\mathcal{D} = \{D_1, D_2 \dots D_n\}$. To guarantee privacy and security, the data owner encrypts the plaintext files into $\xi = \{E_1, E_2, E_3 \dots E_n\}$ and outsource them to a remote cloud server with index $\mathcal{I}$ to support the multi-keyword ranked search where $\mathcal{I}$ is built from the file collections.

To search over the encrypted files ξ, a data user will firstly make a query $\mathcal{Q}$ consisted of the interesting keywords he/she want to search and turn $\mathcal{Q}$ into trapdoor $\mathcal{T}$ using encryption keys. The cloud server will receive $\mathcal{T}$ and search it through an existed index $\mathcal{I}$, then the server will find the matched files and sort them by the matching scores. Furthermore, the cloud server return top-k ID set $\triangle = \{ID_1, ID_2 \dots ID_k\}$. With $\triangle$, the server can easily find the corresponding files, and then return the result $\mathcal{R}$ which includes the required encrypted files.Finally the data user decrypts $\mathcal{R}$ and gets the plaintext files. Since all the procedures are isolated from cloud servers, the cloud server will neither know the keywords the user search nor the exact information of returned files.

The whole framework can be divided into four core algorithms, which is shown as follows:

- **KenGen**
  The data owner generates encryption keys $\mathcal{K}$ for building searchable encryption index and traditional symmetric keys $\mathcal{K}'$ for encrypting and decrypting files. The keys are shared with the owner and users by secure communication protocols.
- **BuildIndex**
  The data owner builds a privacy-protecting index $\mathcal{I}$ supporting multi-keyword ranked search from the database $\mathcal{D}$ based on $\mathcal{K}$ with JL transform. After building the index,

the owner can send both $\mathcal{I}$ and encrypted ξ to cloud server.

- **TrapGen**
  The data user generate the trapdoor $\mathcal{T}$ according to the query $\mathcal{Q}$ including the interesting terms.
- **Search**
  The cloud server compares the $\mathcal{T}$ with each entry in $\mathcal{I}$ and return the matched result $\mathcal{R}$ to the user.

We will talk more in details about how to adopt JL transform to make an efficient and secure index, and how to make a corresponding trapdoor to get most matched results.The following is a set of notations.

B. Notations

- $\mathcal{D}$: the set of total n original plaintext files. $\mathcal{D}$ is denoted as $\mathcal{D} = \{D_1, D_2 \dots D_n\}$.
- ξ: the set of the encrypted files, corresponding to the plain text files $\mathcal{D}$, and ξ is donated as $\xi = \{E_1, E_2, E_3 \dots E_n\}$.
- $\mathcal{W}$: the set of total m keywords in the the dictionary, and $\mathcal{W}$ is presented as $\mathcal{W} = \{W_1, W_2, W_3 \dots W_m\}$.
- $\mathcal{I}$: the secure index $\mathcal{I}$ built by using JL transform. $\mathcal{I}_i$ is corresponding to the file $\mathcal{D}_i$. $\mathcal{I}$ is denoted as $\mathcal{I} = \{I_1, I_2, I_3 \dots I_n\}$.
- $\mathcal{Q}$: the vector which is consisted of the interesting keywords the user want to search.
- $\mathcal{T}$: the trapdoor, built from a query $\mathcal{Q}$, prepared to be sent to the cloud server.
- $\triangle$: the list of the files IDs from returned matched result, and $\triangle$ is denoted as $\triangle = \{ID_1, ID_2 \dots ID_k\}$.

C. Privacy requirements

Generally, the cloud server model is considered as "semi-honest", also called "honest-but-curious" [18]. Specifically, a cloud server will not remove encrypted data files or index from the storage.Besides it will also correctly follow the designated protocol specification and execute the procedure correctly. However, it is curious to infer more data (including trapdoor and index) in the storage and flowing messages to learn additional private information. There are two threat models "**Known Ciphertext Model**" and "**Known Background Model**" [7].

"**Known Ciphertexrt Model**" assumes the cloud server only knows the encrypted files and the index. Since our files and index are both encrypted, without decryption keys, there is no computational possibility for the cloud server to know the exact content of the files.

"**Known Background Model**" means that the cloud server is strong to possess much more knowledge. The server might intentionally collect the statistical information about queries. Because JL transform can leak the Euclidean distance between the two vectors, the cloud server will know the Euclidean distance similarity among the entries in the index. Furthermore, the cloud server might have the capacity to infer which files contains a certain keyword through calculating and guessing the correlation relationship of search requests. However, the server will almost never infer exact frequency of each term in the encrypted index.

In our scheme, the files are protected by AES encryption, which can be considered secure. We focus on privacy of the index and trapdoor. And we expect the index and trapdoor should leak as less information as possible. In MRSE, the author protects the privacy by multiplying two large matrix. In our scheme, we introduce a more efficient scheme – JL transform to encrypt the index. Furthermore we ensure the trapdoor privacy by guarantee that the same query generate different trapdoors at different time, thus the cloud server will be unable to deduce the relationship between trapdoors. This property is also defined as "**trapdoor unlinkability**".

III. PROPOSED SCHEME

In this section, we will talk about the creativity, efficiency and security of JL transform based scheme (J-L scheme for short) in detail.

A. Overview

In our J-L scheme, different from the MRSE and MKQE schemes, which use "inner product similarity" to quantitatively evaluate the coordinate matching, we use the Euclidean distance to evaluate the matching scores of two vectors. In MRSE, the author calculates a secure index which is consisted of a set of vectors, and each vector is corresponding to each file based on the keywords the file contains. Each vector has m bits, m is the total number of keywords in the dictionary. But when we consider the situation that files are stored in the cloud, MRSE will confront the huge space consume problem and this disadvantage will prevent MRSE from being applied to the practical industry use. To solve this problem, in our scheme, each vector also presents each file, but we can reduce the dimension of a vector significantly by applying JL transform. However, we would face another challenge about how to compare the similarity of two vectors using the Euclidean distance after the dimensionality reduction. We then proposed a novel method called "**Optimized Maximum Query**" to generate an efficient trapdoor. Such trapdoor can return top-k matched results more accurately.

B. Johnson-Lindenstrauss transform

Our key idea of dimensionality reduction rises from Johnson-Lindenstrauss lemma: "If points in a vector space are projected onto a randomly selected subspace of suitable high dimension, then the distance between the points are approximately preserved." This random projection has been shown to have promising theoretical properties because using JL transform can keep the accuracy as the original method and at the same time reduce the dimensionality. The detailed Johnson-Lindenstrauss lemma is as follows:

Lemma 1. *Given $\epsilon > 0$ and an integer n, let b be a another positive integer to satisfy $b \geqslant b_0 = O(\epsilon_{-2} logn)$. Consider every $\mathbf{u}, \mathbf{v}$ of n points that $\mathbf{u}, \mathbf{v} \in \mathbb{R}^m$, there exists a map function $f : \mathbb{R}^m \to \mathbb{R}^b$, where $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^b$:*

$$(1-\epsilon)\| \mathbf{u} - \mathbf{v} \|^2 \leq \| f(\mathbf{u}) - f(\mathbf{v}) \|^2 \leq (1+\epsilon)\| \mathbf{u} - \mathbf{v} \|^2$$

This lemma has proven that any set of n points in the m-dimensional Eucildean space can be embedded into b-dimensional Eucildean space, where b is logarithmic in n and independent of m. Furthermore the privacy via JL transform is discussed detail elaboratively in [17]. K. Kenthap *et al* [17] concentrate on the problem whether it is possible for party A to publish some information about each user so that party B can estimate the distance between users without being able to infer any private bit of a user. Their method, which involves projecting each users representation into a random, lower-dimensional space via a sparse JL transform and then adding Gaussian noise to each entry of the lower-dimensional representation, can preserve different degree of privacy where more privacy is desired, the larger the variance of the Gaussian noise is.

In this paper, we will combine JL transform with randomized matrix to meet best privacy request. Then in order to get the best performance, the constraint of b must satisfy : $b \geqslant b_0 = O(\epsilon_{-2} logn)$. This random projection from m-dimensions to b-dimensions can be regarded as a linear transformation represented by $m \times b$ matrix $\mathbf{R}$. Choice of the random matrix $\mathbf{R}$ is our key interest. In order to satisfy the Lemma 1, the $\mathbf{R}$ must hold the following constraints:

- The columns of the random matrix $\mathbf{R}$ are composed of orthonormal vectors with unit length
- The element $r_{i,j}$ in $\mathbf{R}$ have zero mean and unit variance.

1) choosing the projection matrix $\mathbf{R}$*:* There are many ways to choose a suitable projection matrix $\mathbf{R}$ for the dimensionality reduction, and these ways all depend on the properties of the data that need to to preserved. Our choice of the matrix is guided by the two main factors: efficiency and security.

- Each entry of the matrix $\mathbf{R}$ is chosen independently from a Normal distribution with the mean 0 and $\sigma^2 = \frac{1}{b}$, where σ is standard deviation.
- Each entry of the matrix $\mathbf{R}$ is chosen independently and uniformly from $\{-\frac{1}{\sqrt{b}}, +\frac{1}{\sqrt{b}}\}$ randomly.
- Each entry of the matrix is chosen independently to be $\{-\sqrt{\frac{3}{b}}, 0, +\sqrt{\frac{3}{b}}\}$ with corresponding probability $\{\frac{1}{6}, \frac{2}{3}, \frac{1}{6}\}$ respectively.
- The sparse projection matrix of Dasgupta et al. [16] .

In our J-L scheme, we choose the first method as our standard one to generate the projection matrix $\mathbf{R}$ and we will also add the random matrix into the index $\mathcal{I}$ to enhance the security.

C. Making efficient and secure index

The goal of building the index is generating an algorithm to transform the original plaintext index into a privacy-preserving sketch and recover distances between original ones from transformed sketches.

Our algorithms can be expressed as follows: first of all, we build a plaintext index from a set of files. During building the index, we consider the term weight which is not included in MRSE. Instead, we calculate $w(i, j)$ to represent the score weight in j^{th} term in i^{th} document using TF-IDF [19] method. This index is defined as $n \times m$ matrix $\mathbf{P}$ which rows are corresponding to files and columns are corresponding to keywords listed in the dictionary. $w(i, j)$ can be denoted as below:

$$w(i,j) = (1 + log(TF_{i,j}))log\frac{n}{DF_j} \tag{1}$$

$TF_{i,j}$ stands for the frequency number of j^{th} term in i^{th} document, n stands for the total number of documents and DF_j represents the number of documents including the j^{th} term. By adopting TF-IDF method, each position in the index will contain more weight information. That is an improvement compared with MRSE that each position only is occupied by 0 or 1 .

But considering the real-world database, the $\mathbf{P}$ is rather sparse because each file will only include a very small proportion keywords in the whole keywords dictionary. Therefore, there will plenty of zeros in the index. It is essential for us to introduce the random disturbance to covers the positions which are embedded by zeros. We use Ψ as the noise matrix which is selected from a uniform distribution with the arrange from $(0, \gamma]$, where γ is the maximum in the range. And we plus it to the plaintext index $\mathbf{P}$. Secondly, we generate the projection JL transform matrix $\mathbf{R}$ in which each entry of the matrix $\mathbf{R}$ is chosen independently from a normal distribution with mean 0 and $\sigma^2 = \frac{1}{b}$. Then, we get the secure index $\mathcal{I}$ by multiply two matrixes. The details are shown in Alg.1.

Intuitively, given the level of desired privacy, the dimension b will affect the performance and utility of our J-L scheme significantly. On the one hand, if b gets smaller, we will have a lower space consume, but the distance between two vectors will have more deviations after JL transform because the ε_0 will become larger. On the other hand, as b gets larger, the space and computation consume will increase rapidly. Finding the optimal value for the dimension b is challenging and important theoretically. And we will show different b with different experiment results in the next section.

D. Euclidean distance search

Before introducing how to make an efficient trapdoor, we firstly discuss the advantages and disadvantages of using the Euclidean distance to match top-k results. Euclidean distance is unlike MRSE or other methods using "inner product similarity" or "cosine similarity" which is a judgement of orientation and not magnitude. We choose the "cosine similarity" for an example, two vectors with the same orientation will have a cosine similarity of 1, two vectors at angle 90 will have a cosine similarity of 0, and two vectors diametrically opposed have a similarity of -1, which is independent of their magnitude. However, the Euclidean distance is the "ordinary" distance between two points that one would measure with a ruler, and is given by the Pythagorean formula [22]. The Euclidean distance can be regarded as the length of the line segment connecting two points and is affected by the magnitude of the two vectors.

Given two vectors $\vec{p}$ and $\vec{q}$, the Euclidean distance can be presented as:

Algorithm 1 Build The Secure Index

Input:

Plaintext index with frequency whose rows corresponding files and columns corresponding to keywords listed in the dictionary, this index is defined as $n \times m$ matrix $\mathbf{P}$;

Privacy paraments ϵ, σ;

Project dimension b;

Output:

Projection Johnson-Lindenstrauss transform matrix $\mathbf{R}$;

Privacy-preserving encrypted index, which is defined as $n \times b$ matrix $\mathcal{I}$;

1: initialize $\mathbf{R} = \emptyset$;
2: initialize $\mathcal{I} = \emptyset$;
3: generate the projection Johnson-Lindenstrauss transform $m \times b$ matrix $\mathbf{R}$;
4: to protect the privacy and reduce the zeros existed in the $\mathbf{P}$, generate the noise $n \times m$ matrix Ψ ;
5: $\mathbf{P}' = \mathbf{P} + \Psi$;
6: $\mathcal{I} = \mathbf{P}'\mathbf{R}$;
7: construct the $n \times b$ index matrix $\mathcal{I}$;
8: publish($\mathcal{I}, \mathbf{R}$);

$$d(\vec{p}, \vec{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ...(p_m - q_m)^2} = \sqrt{\sum_{i=1}^{m} (p_i - q_i)^2} \quad (2)$$

And the "cosine similarity" can be displayed as follows:

$$d_1(\vec{p}, \vec{q}) = \|\vec{p}\|\|\vec{q}\| cos\theta = \frac{\vec{p} \cdot \vec{q}}{\|\vec{p}\|\|\vec{q}\|} = \frac{\sum_{i=1}^{m} (p_i \times q_i)}{\sqrt{\sum_{i=1}^{m} {p_i}^2} \times \sqrt{\sum_{i=1}^{m} {q_i}^2}} \quad (3)$$

And the inner product similarity can be shown as follows:

$$d_2(\vec{p}, \vec{q}) = \vec{p} \cdot \vec{q} \quad (4)$$

From the equations, we can easily infer that even though the vectors are in the same orientation, their Euclidean distance can also be pretty large because Euclidean distance is easily affected by the magnitude of elements in the vector. We will give a simple example to demonstrate this problem.It is not hard to notice that in the field of multi-keyword ranked search over encrypted database, this is a novel and difficult problem. In order to get high search efficiency, we proposed Optimized Maximum Query to solve this problem.

Table I is a simple example of relevance score index in the palintext. The rows are corresponding to files and the columns are corresponding to the keywords. Let's give a query $\vec{q} = \{0, 1, 0, 1\}$, which means that the second and fourth term is our interesting words.

	w1	w2	w3	w4
$File_1$	1.5	0	1.6	1.2
$File_2$	0	2.5	0	2.4
$File_3$	0	0.4	0	3.2

TABLE I
EXAMPLE OF RELEVANCE SCORE INDEX TABLE

	Inner product	Cosine similarity	Euclidean distance
$File_1$	1.20	0.13	4.80
$File_2$	4.90	0.28	4.21
$File_3$	3.60	0.24	5.20

TABLE II
RESULTS BY DIFFERENT MTHODS

Table II is the results of different methods. The smaller result of Euclidean distance is, the more relative we consider the two vectors are. And it is opposite to measure the relativity by using "inner product similarity" and "cosine similarity", the larger score is then the more relative we consider they are. Through these results, we demonstrate the problems we may confront using the Euclidean distance as the criterion of the results. The vectors of $File_2$ and $File_3$ have the same orientation, but the Euclidean distance between the search query $\vec{q}$ is significantly different. And the distance of $File_1$ from query is smaller than the distance of $File_3$ from query, this is unrealistic because a user always wants the server to return most relative results which will contain all the input keywords respectively. Since the JL transform can only protect the Euclidean distance, we are motivated to propose a novel method called "Optimized Maximum Query" to solve this challenge.

E. Optimized maximum query

In this subsection, we will discuss the method "Optimized Maximum Query" in detail. Speaking of information retrieval in the encrypted database, we have to take two factors (security and efficiency) into consideration. We introduce JL transform to reduce the space consume and enhance the security of the index but we also confront the challenge of making an efficient trapdoor. In the subsection above, we have discussed the shortcomings using the Euclidean distance directly. In order to meet the efficiency demand, we proposed "Optimized Maximum Query" algorithm. The core of our algorithm is to increase the weight of the interesting words in the query vector. To help better understand the algorithm , we take an mathematical analysis based on a given example.

Given a simple search query $\vec{q}$ and a list of first r interesting terms as $\{q_1, q_2...q_r\}$. To enhance the weight of interesting terms in the search query, we expand the search query into m bit length vector by adding zeros behind and multiply the weight parament k to $\vec{q}$. To guarantee the privacy and meet the secure demand of **trapdoor unlinkability** , we will plus a m bit random number vector $\vec{v}$. When compared with an index vector p, we calculate the Euclidean distance to be the matched scores as follows in Equ.5:

$$
\begin{aligned}
score(\vec{p},\vec{q}) &= d^2(\vec{p}, k\times\vec{q}+\vec{\nu}) \\
&= \sum_{i=1}^{m}(p_i - k\times q_i - \nu_i)^2 \\
&= \sum_{j=1}^{r}(p_j - k\times q_j - \nu_j)^2 + \sum_{t=r+1}^{m}(p_t-\nu_t)^2 \\
&= \sum_{j=1}^{r}(p_j^2 - 2kp_jq_j - 2p_j\nu_j + k^2q_j^2 + \nu_j^2) \\
&\quad + \sum_{j=1}^{r}2kq_j\nu_j + \sum_{t=r+1}^{m}(p_t^2 - 2p_t\nu_t + \nu_t^2) \qquad (5)\\
&= \sum_{i=1}^{m}(p_i^2 + \nu_i^2 - 2p_i\nu_i) \\
&\quad + \sum_{j=1}^{r}(k^2q_j^2 - 2kp_jq_j) + \sum_{j=1}^{r}2kq_j\nu_j \\
&= \sum_{i=1}^{m}(p_i-\nu_i)^2 + \sum_{j=1}^{r}k^2q_j^2 - 2k\times(\vec{p}\cdot\vec{q}) \\
&\quad + 2k\times(\vec{q}\cdot\vec{\nu})
\end{aligned}
$$

Since $\vec{q}$ and $\vec{\nu}$ is constant when we make a query and send it to the cloud server to do a search, the score function can be more simplified as follows In Equ.6:

$$
\begin{aligned}
score(\vec{p},\vec{q}) &= \sum_{i=1}^{m}({p_i}^2 + \nu_i^2) - \sum_{i=1}^{m}2p_iv_i + \sum_{j=1}^{r}k^2q_{ij}^2 \\
&\quad - 2k\times(\vec{p}\cdot\vec{q}) + 2k\times(\vec{q}\cdot\vec{\nu}) \\
&= \sum_{i=1}^{m}p_i^2 + \sum_{i=1}^{m}\nu^2 + \sum_{j=1}^{r}k^2q_{ij}^2 - 2\times(\vec{p}\cdot\vec{v}) \qquad (6)\\
&\quad - 2k\times(\vec{p}\cdot\vec{q}) + 2k\times(\vec{q}\cdot\vec{\nu}) \\
&= 2k\times(\vec{q}\cdot\vec{\nu}) - 2k\times(\vec{p}\cdot\vec{q}) - 2\times(\vec{p}\cdot\vec{v}) \\
&\quad + \|p\|^2 + k\|q\|^2 + \|\nu\|^2 \\
&= \mathbf{C} + \|p\|^2 - 2\times(\vec{p}\cdot(k\vec{q}+\vec{v}))
\end{aligned}
$$

In the information retrieval, $\vec{q}$ and $\vec{\nu}$ will change after we input different interesting terms. But when we make one exact search, we can regard $\vec{q}$ and $\vec{\nu}$ are constant variables. Therefore $\mathbf{C}$ is a constant positive number which is related to $\vec{q}$ and $\vec{\nu}$. And $\vec{p}\cdot\vec{q}$ is the inner production similarity. $\|p\|^2$ means the size of the vector $\vec{p}$ and only depends on the construction of documents and keywords. Due to the score equation, to find the most matching documents is the same to find the minimum distance between search query vector and each entry vector in the index.

From the Equ. 6, we can conclude that the search accuracy will be affected by $\|p\|^2$. So how to reduce the influence of $\|p\|^2$ will become a huge challenge. According to the score function, when the weight parament k is large enough and the random vector $\vec{\nu}$ is chosen well enough to make little affect on the result, we can eliminate the influence of $\|p\|^2$ on our final result. In our future study, we will quantity the result using mathematic methods. We assume that $\vec{p}$ is the plaintext query vector with m bits corresponding to one document, and $\vec{i}$ is the encrypted index vector with b bits after the JL transform: $\vec{i} = \vec{p}\,\mathbf{R}$, where $\mathbf{R}$ is $n\times b$ JL transform matrix. The trapdoor vector $\vec{t}$ is from $\vec{t} = (\vec{q}+\vec{v})\mathbf{R}$. Furthermore, since JL transform can maintain the Euclidean distance between two vectors, in our J-L scheme, from Equ.3 we can define the final JL $score_{jl}$ function as below :

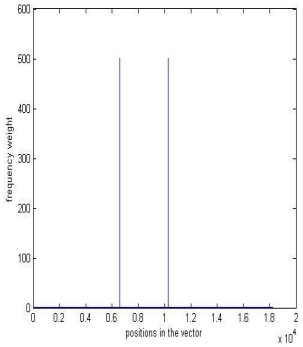

Fig. 2. original query distribution

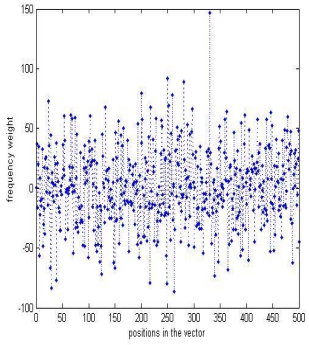

Fig. 3. trapdoor distribution

$$
\begin{aligned}
score_{jl}(\vec{i},\vec{t}) &= d^2(\vec{i},\vec{t}) \\
&= d^2(\vec{p}\mathbf{R}, (k\times\vec{q}+\vec{\nu})\mathbf{R}) \\
&\approx d^2(\vec{p}, k\times\vec{q}+\vec{\nu}) \qquad (7)\\
&= score(\vec{p},\vec{q})
\end{aligned}
$$

Shown in Equ.7, we draw a conclusion that with selectable parameters, our JL scheme results can get the efficiency as using the plaintext Euclidean distance search. And the deviation between plaintext search and encrypted search comes from the parameters of the JL matrix $\mathbf{R}$. Besides, we need less space and time consume than MRSE. We will show the experiment results in the next section. More importantly, we will talk about the security of the trapdoor. Because we use "Optimized Maximum Query" to make a query, the word of the interesting keywords will have large weight which will leak the information of the query vector. In Fig2, the two interesting keywords' weight is much larger than other keywords which is easily attacked by an adversary. However, after the encryption, we generate the trapdoor from the original query, obviously the distribution of query is hidden through JL transform. In Fig.3, the weights in the trapdoor are totally random, and the server could infer little information from the random distribution.

F. Efficient search algorithm

After the analysis of our "Optimized Maximum Query" algorithm to make a trapdoor, in this part, we will talk about how to make an efficient search using our algorithm. The search algorithm can be described simply as follows: given a m bit query vector $\mathcal{Q}$,we first multiply a weight parameter k and then plus a random vector $\vec{\nu}$ as Optimized Maximum

Algorithm 2 Efficient Search

Input:

Encrypted secure index $\mathcal{I}$;
search query vector $\mathcal{Q}$;
the random vector $\vec{\nu}$;
Projection Johnson-Lindenstrauss transform matrix $\mathbf{R}$;

Output:

the list of the files IDs from returning matching result Δ;

1: initialize $\Delta = \emptyset$;
2: initialize trapdoor $\mathcal{T} = \emptyset$;
3: $\mathcal{T} = (\mathcal{Q} + \vec{\nu})\mathbf{R}$;
4: generate the trapdoor $\mathcal{T}$;
5: calculate the $score_{jl}$ between $\mathcal{I}_i$ and $\mathcal{T}$;
6: return the top-k minimum scores results Δ;

Query says. Then we generate the trapdoor $\mathcal{T}$ by multiply it with the Projection Johnson-Lindenstrauss transform matrix $\mathbf{R}$. At last, we compare the Euclidean distance between each entry in the index $\mathcal{I}$ and return the top-k results with low scores. The detail of making efficient search is in Alg.2.

Generally, since the frequency in the trapdoor is hidden and isolated from the cloud server and JL transform is a one-direction function, the cloud server cannot decrypt the trapdoor to obtain the original search query $\mathcal{Q}$. Suppose even the user input the same search query at different time, the distortion of different random vector $\vec{\nu}$ will have influence on generating the trapdoor $\mathcal{T}$, the cloud server will receive two different trapdoors and never know the relationship between the two trapdoors. By adopting these methods, we not only guarantee the search accuracy but also live up to the privacy demand of "trapdoor unlikability".As for "rank privacy",our scheme was unable to avoid the curious server to infer some rank information from the returned results like MRSE,as this is a necessary privacy sacrifice to get better user experiment.We noticed that different b may provide different perturbation on the ranked result,which will be digged in our future work.

IV. Result and Comparison

In this section, we would compare our purposed J-L scheme with MRSE. We use precision rate and recall rate [21] as our criterion to evaluate the search accuracy and compare search results with standard results to calculate precision rate and recall rate. Taking account of all former algorithms, to our best knowledge and considering the fairness, MRSE algorithm is the most representive one,since MKQE will face the space consume problem as MRSE does. By the way, OPE method is not using the "vector space model" like MRSE and our J-L scheme. In a word, we demonstrate a thorough experimental evaluation on the TREC data [20], which is consisted of 7594 documents and 18238 distinct terms. The experiment is implemented by MATLAB language and conducted on a personal computer to imitate the cloud server with i-3 CORE 2.13GHz processor and Window 7 home basic system. In the experiment,we choose the weight parameter $k = 1500$,and the random $\vec{\nu}$ is a uniform random distribution between $\{0,1\}$.

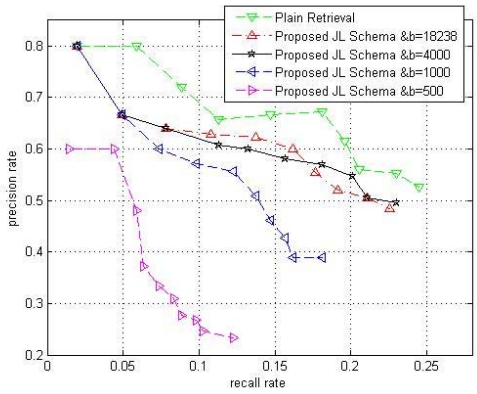

Fig. 4. the search efficiency with different b

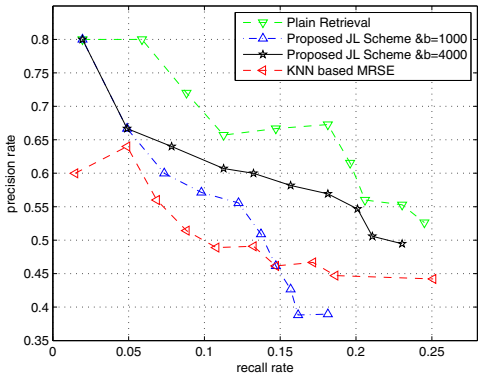

Fig. 5. the search efficiency of different algorithms

The parameter b represents the dimension of columns in the projection JL transform matrix $\mathbf{R}$, it is also the targeted dimension we will reduce the original index vector to. Fig.4 shows different b will have different performance on the research result. As the b becomes larger, we will get the high performance but the space consume will increase at the same time.So choosing the selectable parameter k and b is feasible according to the accuracy and efficiency need of the users.

Compared with MRSE, we choose our $b = 1000$ and $b = 4000$ where the original size is 18238, which means there are 18238 different keywords in the dictionary. According to the results shown in Fig.5, our scheme performs much better than MRSE when returning the top 10-30 results and the results will get much closer to the plaintext as we increase the parameter b. In the real-world circumstance, the user only concentrate on the high ranked results,so our scheme can meet the user's need perfectly and be adopted into practical ways.

Fig.6 shows the space complexity of our algorithms with different parameter b and the KNN-based MRSE. We conduct the logarithmic coordinate systems to measure the size of space complexity. From Fig.6 we can see it is highly impressive that our scheme consumes much less space complexity than

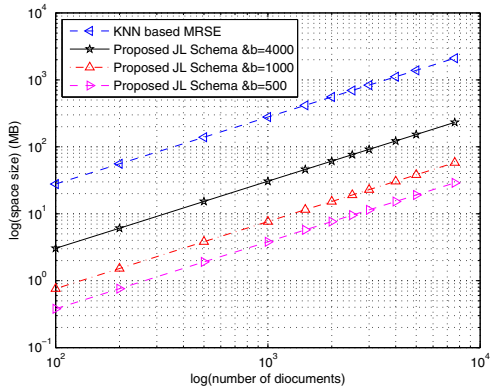

Fig. 6. the Space Complexity

MRSE. In the other words, reduction of the space complexity strongly is one of the most important contribution in this paper.

V. Conclusion

In this paper, we fulfilled a system model to realize a multi-keyword ranked search over the encrypted database in cloud computing. To meet the challenge of guaranteeing efficient retrieval, we proposed our J-L scheme by applying the Johnson-Lindenstrauss transform into this field for the first time. And we proposed "Optimized Maximum Query" to make an efficient trapdoor. Experiment results showed our novel scheme achieves advantages both in accuracy performance and storage space.

We will provide strict mathematic analysis on the chosen random vector of our scheme in our future work on the research. Since the JL transform is the first time introduced into the field of information retrieval over encrypted database, we will do deeper research on this method and hope to expand the application to other related signal processing fields like encrypted image search perfectly.

Acknowledgment

This work was supported in part by the Natural Science Foundation of China under Grant 61170234 and Grant 60803155, by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA06030601, and by the Funding of Science and Technology on Information Assurance Laboratory under Grant KJ-13-02.

References

[1] M. Armbrust,A. Fox,R. Griffith,and A.D. Joseph. A view of cloud computing, Communications of the ACM, 2010, pp. 50-58.
[2] K. Ren, C. Wang, and Q. Wang, Security Challengs for the public Cloud. IEEE InternetComputing, vol.16,no.1, 2012, pp.69-73.
[3] M. Abdalla,M. Bellare,D. Catalano, and et al. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions[C],Advances in CryptologyCCRYPTO. Springer Berlin Heidelberg, 2005 pp.205-222.
[4] Goh, Eu-Jin.Secure indexes.Cryptology ePrint Archive,Report 2003/216, 2003.
[5] R. Curtmola,J. A. Garay, S. Kamara,and R. Ostrovsky, Searchable symmetric encryption: improved definitions and efficient constructions, Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006, pp. 79-88.
[6] Wang C, Cao N, Li J, et al. Secure ranked keyword search over encrypted cloud data[C]//Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on. IEEE, 2010: 253-262.
[7] Cao N, Wang C, Li M, et al. Privacy-preserving multi-keyword ranked search over encrypted cloud data[C]//INFOCOM, 2011 Proceedings IEEE. IEEE, 2011: 829-837.
[8] Z. Xu,W. Kang,R. Li,K. Yow,and C. Xu. Efficient Multi-Keyword Ranked Query on Encrypted Data in the Cloud, Parallel and Distributed Systems (ICPADS), IEEE 18th International Conference on,2012, pp. 244-251
[9] C. Yang,W. Zhang,J. Xu,J. Xu,and N. Yu. A Fast Privacy-Preserving Multi-keyword Search Scheme on Cloud Data, Cloud and Service Computing (CSC), International Conference on. IEEE, 2012, pp. 104-110.
[10] J. Xu,W. Zhang,C. Yang,J. Xu,and N. Yu, Two-Step-Ranking Secure Multi-Keyword Search over Encrypted Cloud Data, Cloud and Service Computing (CSC), International Conference on. IEEE, 2012, pp. 124-130.
[11] D. Song, D. Wagner,and A. Perrig. Practical techniques for searches on encrypted data, in Proc. of IEEE Symposium on Security and Privacy00, 2000, pp. 44-55.
[12] National Bureau of Standards, Data Encryption Standard.U.S. Department of Commerce, Washington, DC, USA, Jan. 1977.
[13] H. Dobbertin, V. Rijmen, and A. Sowa, Advanced Encryption StandardC AES: 4th International Conference, AES 2004, Bonn, Germany, May 10-12, 2004 : Revised Selected and Invited Papers. Lecture Notes in Computer Science, Springer, 2005.
[14] Microsoft azure. http://www.windowsazure.com
[15] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz maps into a Hilbert space. InContemp Math, volume 26, 1984, pp. 189-206.
[16] A. Dasgupta, R. Kumar, and T. Sarlos. A sparse Johnson-Lindenstrauss transform. In STOC, 2010, pp. 341C350.
[17] a Kenthapadi, K., Korolova, A., Mironov, I.,Mishra, N. (2012). Privacy via the johnson-lindenstrauss transform. arXiv preprint arXiv:1204.2606.
[18] S. Yu, C. Wang, K. Ren, and W. Lou, Achieving secure, scalable, and fine-grained data access control in cloud computing, in Proc. of INFOCOM , 2010.
[19] S. E. Robertson and K. S. Jones, Simple Proven Approaches to Text Retrieval. Thchenical Report TR356, Cambridge University Computer Laboratory,1997.
[20] P. Bailey,N. Craswell,I. Soboroff,and A. P. Vries, The CSIRO Enterprise Search Test Collection, SIGIR Forum 41(2), 2007, pp. 42-45.
[21] http://en.wikipedia.org/wiki/Precision_ and_ recall.
[22] http://en.wikipedia.org/wiki/Pythagorean_theorem.