IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 1, Issue 1, March, 2013
ISSN: 2320 - 8791
www.ijreat.org

Blob Repositised Metatag Indexed Crawler In Cloud Servers

Surekha[1], Manju[2], Samundeeswari[3] and Mrs.Shanthi[4] (Guide)
Computer Science and Engineering, Veltech Hightech Dr.Rangarajan Dr.Sakunthala Engineering College, Chennai-600062, Tamilnadu, India,

ABSTRACT
Ranked search greatly enhances system usability by enabling search result relevance ranking instead of sending undifferentiated results, and further ensures the file retrieval accuracy.The statistical measure approach, i.e., relevance score, from information retrieval to build a secure searchable index, and develop a one-to-many order-preserving mapping technique to properly protect those sensitive score information.The resulting design is able to facilitate efficient server-side ranking without losing keyword privacy. Thorough analysis shows that our proposed solution enjoys "as strong-as-possible" security guarantee compared to previous searchable encryption schemes, while correctly realizing the goal of ranked keyword search.Extensive experimental results demonstrate the efficiency of the proposed solution.

keywords—Ranked keyword search, relevance score, searchable encryption, order-preserving mapping.

1.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 [2]. The benefits brought by this new computing model include but are not limited to: relief of the burden for storage manage- ment, universal data access with independent geographical locations, and avoidance of capital expenditure on hard-ware, software, and personnel maintenances, etc., [3].

As Cloud Computing becomes prevalent, more and more sensitive information are being centralized into the cloud, such as e-mails, personal health records, company finance data, and government documents, etc. The fact that data owners and cloud server are no longer in the same trusted domain may put the outsourced unencrypted data at risk [4], [33]: the cloud server may leak data information to unauthorized entities [5] or even be hacked [6]. It follows that sensitive data have to be encrypted prior to outsourcing for data privacy and combating unsolicited accesses. However, data encryption makes effective data utilization a very challenging task given that there could be a large amount of outsourced data files. Besides, in Cloud Computing, data owners may share their outsourced data with a large number of users, who might want to only retrieve certain specific data files they are interested in during a given session. One of the most popular ways to do so is through keyword-based search. Such keyword search technique allows users to selectively retrieve files of interest and has been widely applied in plaintext search scenarios [7]. Unfortunately, data encryption, which restricts user's ability to perform keyword search and further demands the protection of keyword privacy, makes the traditional plaintext search methods fail for encrypted cloud data.

Although traditional searchable encryption schemes (e.g., [8], [9], [10], [11], [12], to list a few) allow a user to securely search over encrypted data through keywords without first decrypting it, these techniques support only conventional Boolean keyword search,[1] without capturing any relevance of the files in the search result. When directly applied in large collaborative data outsourcing cloud environment, they may suffer from the following two main drawbacks. On the one hand, for each search request, users without preknowledge of the encrypted cloud data have to go through every retrieved file in order to find ones most matching their interest, which demands possibly large amount of postprocessing overhead; On the other hand, invariably sending back all files solely based on presence/ absence of the keyword further incurs large unnecessary

1. *In the existing symmetric key-based searchable encryption schemes, the support of disjunctive Boolean operation (OR) on multiple keywords searches still remains an open problem.*

network traffic, which is absolutely undesirable in today's pay-as-you-use cloud paradigm. In short, lacking of effective mechanisms to ensure the file retrieval accuracy is a significant drawback of existing searchable encryption schemes in the context of Cloud Computing. Nonetheless, the state of the art in information retrieval (IR) community has already been utilizing various scoring mechanisms [13] to quantify and rank order the relevance of files in response to any given search query. Although the importance of ranked search has received attention for a long history in the context of plaintext searching by IR community, surprisingly, it is still being overlooked and remains to be addressed in the context of encrypted data search.

Therefore, how to enable a searchable encryption system with support of secure ranked search is the problem tackled in this paper. Our work is among the first few ones to explore ranked search over encrypted data in Cloud Computing. Ranked search greatly enhances system usability by returning the matching files in a ranked order regarding to certain relevance criteria (e.g., keyword frequency), thus making one step closer toward practical deployment of privacy-preserving data hosting services in the context of Cloud Computing.

To achieve our design goals on both system security and usability, we propose to bring together the advance of both crypto and IR commu- nity to design the ranked searchable symmetric encryption (RSSE) scheme, in the spirit of "as-strong-as-possible" security guarantee. Specifically, we explore the statistical measure approach from IR and text mining to embed weight information (i.e., relevance score) of each file during the establishment of searchable index before outsourcing the encrypted file collection.

As directly outsourcing relevance scores will leak lots of sensitive frequency information against the keyword privacy, we then integrate a recent crypto primitive [14] order-preserving symmetric encryption (OPSE) and properly modify it to develop a one- to-many order-preserving mapping technique for our purpose to protect those sensitive weight information, while providing efficient ranked search functionalities. Our contribution can be summarized as follows:

1. For the first time, we define the problem of secure ranked keyword search over encrypted cloud data, and provide such an effective protocol, which fulfills the secure ranked search functionality with little relevance score information leakage against keyword privacy.
2. Thorough security analysis shows that our ranked searchable symmetric encryption scheme indeed enjoys "as-strong-as-possible" security guarantee compared to previous searchable symmetric encryption (SSE) schemes.
3. We investigate the practical considerations and enhancements of our ranked search mechanism, including the efficient support of relevance score dynamics, the authentication of ranked search results, and the reversibility of our proposed one-to-many order-preserving mapping technique.
4. Extensive experimental results demonstrate the effectiveness and efficiency of the proposed solution.

The rest of the paper is organized as follows: Section 2 gives the system and threat model, our design goals, notations, and preliminaries. Then, we provide the frame- work, definitions, and basic scheme in Section 3, followed by Section 4, which gives the detailed description of our ranked searchable symmetric encryption system.. Section 5 studies further enhancements and practical considerations, followed by Section 6 gives the concluding remark of the whole paper. Finally section 7 gives the references of papers,literature reviews and links

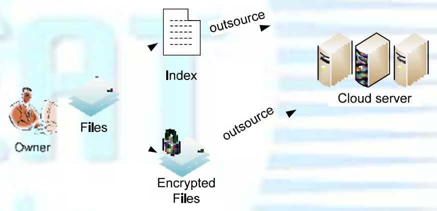

Fig. 1. Architecture for search over encrypted cloud data.

2 PROBLEM STATEMENT

2.1The System and Threat Model

We consider an encrypted cloud data hosting service involving three different entities, as illustrated in Fig. 1: data owner, data user, and cloud server. Data owner has a collection of n data files C =(F_1 ; F_2 ; . . . ; F_n)(that he wants to outsource on the cloud server in encrypted form while still keeping the capability to search through them for effective data utilization reasons. To do so, before outsourcing, data owner will first build a secure searchable index **I** from a set of m distinct keywords W=(w_1 ; w_2 ; . . . ; w_m) extracted^2 from the file collection C, and store both the index **I** and the encrypted file collection C on the cloud server.

We assume the authorization between the data owner and users is appropriately done. To search the file collection for a given keyword w, an authorized user generates and submits a search request in a secret form—a trapdoor T_w of the keyword w—to the cloud server. Upon receiving the search request T_w, the cloud server is responsible to search the index **I** and return the corresponding set of files to the user. We consider the secure ranked keyword search problem as follows: the search result should be returned according to certain ranked relevance criteria (e.g., keyword frequency-based scores, as will be introduced shortly), to improve file retrieval accuracy for users without prior knowledge on the file collection C. However, cloud server should learn nothing or little about the relevance criteria as they exhibit significant sensitive information against key- word privacy. To reduce bandwidth, the user may send an optional value k along with the trapdoor T_W and cloud server only sends back the top-k most relevant files to the user's interested keyword w.

We primarily consider an "honest-but-curious" server in our model, which is consistent with most of the previous searchable encryption schemes. We assume the

2.2Design and Goals

To enable ranked searchable symmetric encryption for effective utilization of outsourced and encrypted cloud data under the aforementioned model, our system design should achieve the following security and performance guarantee. Specifically, we have the following goals: 1) Ranked keyword search: to explore different mechanisms for designing effective ranked search schemes based on the existing searchable encryption framework; 2) Security guarantee: to prevent cloud server from learning the plaintext of either the data files or the searched keywords, and achieve the "as-strong-as-possible" security strength compared to existing searchable encryption schemes; 3) Efficiency: above goals should be achieved with minimum communication and computation overhead.

2.3Notation and Preliminaries

C—the file collection to be outsourced, denoted as a set of n data files files C =(F_1 ; F_2 ; . . . ; F_n).

W —the distinct keywords extracted from file collection C, denoted as a set of m words W=(w_1 ; w_2 ; . . . ; w_m) .

Id(F_j)—the identifier of file F_j

I—searchable index .

F (w_i)—the set of identifiers of files thetrapdoor generated by a user as a search request .

F (w_i)—the set of identifiers of files

cloud server acts in an "honest" fashion and correctly follows the

2. To reduce the size of index, a list of standard IR techniques can be adopted, including case folding, stemming, and stop words, etc. We omit this process of keyword extraction and refinement and refer readers to [7] for more details.

designated protocol specification, but is "curious" to infer and analyze the message flow received during the protocol so as to learn additional information. In other words, the cloud server has no intention to actively modify the message flow or disrupt any other kind of services. However, in some unexpected events, the cloud server may behave beyond the "honest-but-curious" model. We specifically deal with this scenario in Section 6.2.

TABLE 1
An Example Posting List of the Inverted Index

Word	w_i				
File ID	F_{i_1}	F_{i_2}	F_{i_3}	$\cdots$	$F_{i_{N_i}}$
Relevance Score	6.52	2.29	13.42	4.76	13.80

We now introduce some necessary information retrieval background for our proposed scheme:

Inverted index. In information retrieval, inverted index

(also referred to as postings file) is a widely used indexing structure that stores a list of mappings from keywords to the corresponding set of files that contain this keyword, allowing full text search [13]. For ranked search purposes, the task of determining which files are most relevant is typically done by assigning a numerical score, which can be precomputed, to each file based on some ranking function introduced below. One example posting list of an index is shown in Table 1. We will use this inverted index structure to give our basic ranked searchable symmetric encryption construction.

Ranking function. In information retrieval, a ranking

function is used to calculate relevance scores of matching files to a given search request. The most widely used

$$Score(Q, F_d) = \sum_{t \in Q} \frac{1}{|F_d|} \cdot (1 + \ln f_{d,t}) \cdot \ln\left(1 + \frac{N}{f_t}\right). \quad (1)$$

statistical measurement for evaluating relevance score in the information retrieval community uses the TF IDF rule, where term frequency (TF) is simply the number of times a given term or keyword (we will use them interchangeably hereafter) appears within a file (to measure the importance of the term within the particular file), and inversedocumentfrequency(IDF) is obtained by dividing the number of files in the whole collection by the number of files containing the term (to measure the overall importance of the term within the whole collection). Here, Q denotes the searched keywords; $f_{d;t}$ denotes the TF of term t in file F_d ; f_t denotes the number of files that contain term t;

N denotes the total number of files in the collection; and $|F_d|$ is the length of file F_d, obtained by counting the number of indexed terms, functioning as the normalistion factor.

3 THE DEFINITIONS AND BASIC SCHEME

In the introduction, we have motivated the r anked keyword search over encrypted data to achieve economies of scale for Cloud Computing. In this section, we start from the review of existing searchable symmetric encryption schemes and provide the definitions and framework for our proposed ranked searchable symmetric encryption. Note that by following the same security guarantee of existing SSE, it would be very inefficient to support ranked search functionality over encrypted data, as demonstrated in our basic scheme. The discussion of its demerits will lead to our proposed scheme.

3.1 Background on Searchable Symmetric Encryption

Searchable encryption allows data owner to outsource his data in an encrypted manner while maintaining the selectively search capability over the encrypted data. Generally, searchable encryption can be achieved in its full functionality using an oblivious RAMs [16]. Although hiding everything during the search from a malicious server (including access pattern), utilizing oblivious RAM usually brings the cost of logarithmic number of interac- tions between the user and the server for each search request. Thus, in order to achieve more efficient solutions, almost all the existing works on searchable encryption literature resort to the weakened security guarantee, i.e., revealing the access pattern and search pattern but nothing else. Here, access pattern refers to the outcome of the search result, i.e., which files have been retrieved. The search pattern includes the equality pattern among the two search requests (whether two searches were performed for the same keyword), and any information derived thereafter from this statement. We refer readers to [12] for the thorough discussion on SSE definitions.

Having a correct intuition on the security guarantee of existing SSE literature is very important for us to define our ranked searchable symmetric encryption problem. As later, we will show that following the exactly same security guarantee of existing SSE scheme, it would be very inefficient to achieve ranked keyword search, a which motivates us to further weaken the security guarantee of existing SSE appropriately (leak the relative relevance order but not the relevance score) and realize an “as-strong-as- possible” ranked searchable symmetric encryption. Actu- ally, this notion has been employed by cryptographers in many recent work [14], [17] where efficiency is preferred over security.

3.2 Definitions and Framework of RSSE System

We follow the similar framework of previously proposed searchable symmetric encryption schemes [12] and adapt the framework for our ranked searchable encryption system. A ranked searchable encryption scheme consists of four algorithms (KeyGen, BuildIndex, TrapdoorGen, SearchIndex). Our ranked searchable encryption system can be constructed from these four algorithms in two phases, Setup and Retrieval:

Setup. The data owner initializes the public and secret parameters of the system by executing Key- Gen, and pre-processes the data file collection C by using BuildIndex to generate the searchable index from the unique words extracted from C. The owner then encrypts the data file collection C, and publishes the index including the keyword frequency-based relevance scores in some encrypted form, together with the encrypted collection C to the Cloud. As part of Setup phase, the data owner also needs to distribute secret parameters) to a group of authorized users by employing off-the-shelf public key cryptography or more efficient primitive such as broadcast encryption.

Retrieval. The user uses TrapdoorGen to generate a secure trapdoor corresponding to his interested keyword, and submits it to the cloud server.Upon receiving the trapdoor, the cloud server will derive a list of matched file IDs and their corre- sponding encrypted relevance scores by searching the index via SearchIndex. The matched files should be sent back in a ranked sequence based on the relevance scores. However, the server should learn nothing or little beyond the order of the relevance scores.

4 EFFICIENT RANKED SEARCHABLE SYMMETRIC ENCRYPTION SCHEME

The above straightforward approach demonstrates the core problem that causes the inefficiency of ranked searchable encryption. That is how to let server quickly perform the ranking without actually knowing the relevance scores. To effectively support ranked search over encrypted file collection, we now resort to the newly developed crypto- graphic primitive—order preserving symmetric encryption [14] to achieve more practical performance. Note that by resorting to OPSE, our security guarantee of RSSE is inherently weakened compared to SSE, as we now let server know the relevance order. However, this is the information we want to trade off for efficient RSSE, as
discussed in previous Section 3. We will first briefly discuss the primitive of OPSE and its pros and cons. Then, we show how we can adapt it to suit our purpose for ranked searchable encryption with an “as-strong-as-possible” se- curity guarantee. Finally, we demonstrate how to choose different scheme parameters via concrete examples.

4.1Using Order Preserving Symmetric Encryption The OPSE is a deterministic encryption scheme where the numerical ordering of the plaintexts gets preserved by the encryption function. Boldyreva et al. [14] gives the first cryptographic study of primitive and provides a construction that is provably secure under the security framework of pseudorandom function or pseudorandom permutation. Namely, considering that any order-preser- ving function gð Þ from domain D ={1; . . . ; M } to range R ={1; . . . ; N} can be uniquely defined by a combination of M out of N ordered items, an OPSE is then said to be secure if and only if an adversary has to perform a brute force search over all the possible combinations of M out of N to break the encryption scheme. If the security level is chosen to be 80 bits, then it is suggested to choose M = N/2 > 80 so that the total number of combinations will be greater than 2^{80}. Their construction is based on an uncovered relationship between a random order-preserving function (which meets the above security notion) and the hypergeometric probability distribution, which will later be denoted as HGD. We refer readers to [14] for more details about OPSE and its security definition.At the first glance, by changing the relevance score encryption from the standard indistinguishable symmetric encryption scheme to this OPSE, it seems to follow directly that efficient relevance score ranking can be achieved just like in the plaintext domain. However, as pointed out earlier, the OPSE is a deterministic encryption scheme. This inherent deterministic property, if not treated appropri- ately, will still leak a lot of information as any deterministic encryption scheme will do. One such information leakage is the plaintext distribution. Take Fig. 2, for example, which shows a skewed relevance score distribution of keyword “network,” sampled from 1,000 files of our test collection. For easy exposition, we encode the actual score into 128 levels in domain from 1 to 128. Due to the deterministic property, if we use OPSE directly over these sampled relevance scores, the resulting ciphertext shall share exactly the same distribution as the relevance score in Fig. 2. On the other hand, previous research works [18], [22] have shown that the score distribution can be seen as keyword specific. Specifically, in [22], the authors have shown that the TF distribution of certain keywords from the Enron e-mail corpus[3] can be very peaky, and thus result in significant information leak for the corresponding keyword. In [18], the authors further point out that the TF distribution of the keyword in a given file collection usually follows a power law distribution, regardless of the popularity of the key- word. Their results on a few test file collections show that not only different keywords can be differentiated by the slope and value range of their TF distribution, but even the normalized TF distributions, i.e., the original score distribu- tions (see (2)), can be keyword specific. Thus, with certain background information on the file collection, such as knowing it contains only technical research papers, the adversary may be able to reverse engineer the keyword “network” directly from the encrypted score distribution without actually breaking the trapdoor construction, nor does the adversary need to break the OPSE.

4.2 One-to-Many Order-Preserving Mapping

Therefore, we have to modify the OPSE to suit our purpose. In order to reduce the amount of information leakage from the deterministic property, an one-to-many OPSE scheme is thus desired, which can flatten or obfuscate the original relevance score distribution, increase its randomness, and still preserve the plaintext order. To do so, we first briefly review the encryption process of original deterministic OPSE, where a plaintext m in domain D is always mapped to the samerandom-sized nonoverlapping interval bucket in range R, determined by a keyed binary search over the range R and the result of a random HGD sampling function. A ciphertext c is then chosen within the bucket by using m as the seed for some random selection function. Our one-to-many order-preserving mapping employs the random plaintext-to-bucket mapping of OPSE, but incorpo- rates the unique file IDs together with the plaintext m as the random seed in the final ciphertext chosen process. Due to the use of unique file ID as part of random selection seed, the same plaintext m will no longer be deterministically assigned to the same ciphertext c, but instead a random value within the randomly assigned bucket in range R. The whole process is shown in Algorithm 1, adapted from [14].TapGen is the random coin generator and HYGEINV is the he efficient function implemented in Matlab as our instance for the HGDð_Þ sampling function. The correctness of our one-to-many order-preserving mapping follows directly from the Algorithm 1. Note that our rational is to use the OPSE block cipher as a tool for different application scenarios and achieve better security, which is suggested by and consistent with [14]. Now, if we denote OPMas our one-to-many order-preserving mapping function with parameter: OPM:$\{0.1\}^{l}\{0.1\}^{\log|D|} \rightarrow \{0.1\}^{\log|R|}$,our proposed
RSSE scheme can be described as follows:
In the setup phase

Algorithm 1. One-to-Many Order-Preserving Mapping-$\mathcal{OPM}$

```
1:  procedure $\mathcal{OPM}_K(\mathcal{D}, \mathcal{R}, m, \mathrm{id}(F))$
2:      while $|\mathcal{D}|! = 1$ do
3:          $\{\mathcal{D}, \mathcal{R}\} \leftarrow$ BinarySearch$(K, \mathcal{D}, \mathcal{R}, m)$;
4:      end while
5:      $coin \xleftarrow{R}$ TapeGen$(K, (\mathcal{D}, \mathcal{R}, 1||m, \mathrm{id}(F)))$;
6:      $c \xleftarrow{coin} \mathcal{R}$;
7:      return $c$;
8:  end procedure

9:  procedure BinarySearch$(K, \mathcal{D}, \mathcal{R}, m)$;
10:     $M \leftarrow |\mathcal{D}|$; $N \leftarrow |\mathcal{R}|$;
11:     $d \leftarrow \min(\mathcal{D}) - 1$; $r \leftarrow \min(\mathcal{R}) - 1$;
12:     $y \leftarrow r + \lceil N/2 \rceil$;
13:     $coin \xleftarrow{R}$ TapeGen$(K, (\mathcal{D}, \mathcal{R}, 0||y))$;
14:     $x \xleftarrow{R} d +$ HYGEINV$(coin, M, N, y - r)$;
15:     if $m \leq x$ then
16:         $\mathcal{D} \leftarrow \{d + 1, \ldots, x\}$;
17:         $\mathcal{R} \leftarrow \{r + 1, \ldots, y\}$;
18:     else
19:         $\mathcal{D} \leftarrow \{x + 1, \ldots, d + M\}$;
20:         $\mathcal{R} \leftarrow \{y + 1, \ldots, r + N\}$;
21:     end if
22:     return $\{\mathcal{D}, \mathcal{R}\}$;
23: end procedure
```

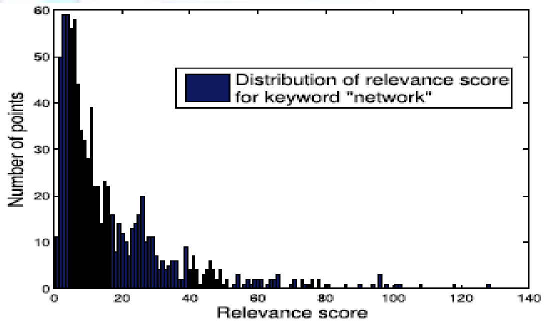

Fig. 2. An example of relevance score distribution.

4.3Authenticating Ranked Search Result

In practice, cloud servers may sometimes behave beyond the semihonest model. This can happen either because cloud server intentionally wants to do so for saving cost when handling large number of search requests, or there may be software bugs, or internal/external attacks. Thus, enabling a search result authentication mechanism that can detect such unexpected behaviors of cloud server is also of practical interest and worth further investigationTo authenticate a ranked search result (or Top-k retrieval), one need to ensure: 1) the retrieved results are the most relevant ones; 2) the relevance sequence among the results are not disrupted. To achieve this two authentication requirements, we propose to utilize the one way hash chain technique, which can be added directly on top of the previous RSSE design. Let H ð Þ denote some cryptographic one-way hash function, such as SHA-1. Our mechanism requires one more secret value u in the Setup phase to be generated and shared between data owner and users.

4.4 SupportingScore Dynamics

In Cloud Computing, outsourced file collection might not only be accessed but also updated frequently for various application purposes (see [19], [20], [21], for example). Hence, supporting the score dynamics in the searchable index for an RSSE system, which is reflected from the corresponding file collection updates, is thus of practical importance. Here, we consider score dynamics as adding newly encrypted scores for newly created files, or modify- ing old encrypted scores for modification of existing files in the file collection. Ideally, given a posting list in the inverted index, the encryption of all these newly changed scores should be incorporated directly without affecting the order of all other previously encrypted scores, and we show that our proposed one-to-many order-preserving mapping does exactly that. Note that we do not consider file deletion scenarios because it is not hard to infer that deleting any file and its score does not affect the ranking orders of the remaining files in the searchable index.This graceful property of supporting score dynamics is inherited from the original OPSE scheme, even though we made some adaptations in the mapping process. This can be observed from the BinarySearchð Þ procedure in Algorithm 1, where the same score will always be mapped to the same random-sized nonoverlapping bucket, given the same encryption key and the same parameters of the plaintext domain D and ciphertext range R. Because the buckets themselves are nonoverlapping, the newly changed scores indeed do not affect previously mapped values. Thus, with this property, the data owner can avoid the recomputation of the whole score encryption for all the file collection, but instead just handle those changed scores whenever neces- sary. Note that the scores chosen from the same bucket are treated as ties and their order can be set arbitrarily.

5.PERFORMANCE ANALYSIS

The performance of our scheme is evaluated regarding the effectiveness and efficiency of our proposed one-to-many order-preserving mapping, as well as the overall performance of our RSSE scheme, including the cost of index construction as well as the time necessary for searches.

5.1 Efficiency of Search

The search time includes fetching the posting list in the index, decrypting, and rank ordering each entries. Our focus is on top-k retrieval. As the encrypted scores are order preserved, server can process the top-k retrieval almost as fast as in the plaintext domain. Note that the server does not have to traverse every posting list for each given trapdoor, but instead uses a tree-based data structure to fetch the corresponding list. Therefore, the overall search time cost is almost as efficient as on unencrypted data. Fig. 3 list our search time cost against the value of k increases, for the same index constructed below.

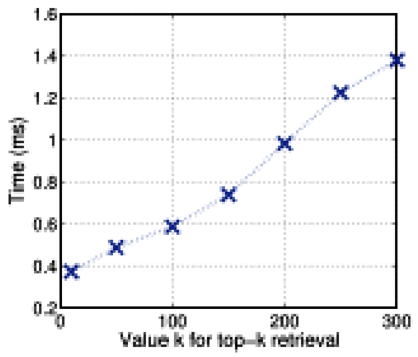

Fig 2: The time cost for top-k retrieval

6.FUTURE ENHNCEMENT

We investigate future enhancement of our ranked search mechanism including the reversibility of our proposed one-to-many order-preserving mapping technique.

7.CONCLUSION

In this paper, as an initial attempt, we motivate and solve the problem of supporting efficient ranked keyword search for achieving effective utilization of remotely stored encrypted data in Cloud Computing.We then appropriately weaken the security guarantee, resort to the newly developed crypto primitive OPSE, and derive an efficient one-to-many order- preserving mapping function, which allows the effective RSSE to be designed. We have done enhancements of our ranked search mechanism, including the efficient support of relevance score dynamics, the authentication of ranked search results, Through thorough security analysis, we show that our proposed solution is secure and privacy preser- ving, while correctly realizing the goal of ranked keyword search.Extensive experimental results demonstrate the efficiency of our solution.

8.REFERENCES

[1] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure Ranked Keyword Search over Encrypted Cloud Data," Proc. IEEE 30th Int'l Conf. Distributed Computing Systems (ICDCS '10), 2010.
[2] P. Mell and T. Grance, "Draft Nist Working Definition of Cloud Computing,"
[3] "Above the Clouds: A Berkeley View of Cloud Comput- ing,"
[4] Cloud Security Alliance "Security Guidance for Critical Areas of Focus in Cloud Computing,"
[5] Z. Slocum, "Your Google Docs: Soon in Search Results?" http://news.cnet.com/8301-17939_109-10357137-2.html, 2009.
[6] B. Krebs, "Payment Processor Breach May Be Largest Ever," h t tp:// voices.washin gtonp ost .co m /securit yfix/2 009/01/ payment_processor_breach_may_b.html, Jan. 2009.
[7] I.H. Witten, A. Moffat, and T.C. Bell, Managing Gigabytes: Compressing and Indexing Documents and Images. Morgan Kauf- mann, May 1999
[8] D. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," Proc. IEEE Symp. Security and Privacy, 2000.
[9] E.-J. Goh, "Secure Indexes," Technical Report 2003/216, Cryptology ePrint Archive, http://eprint.iacr.org/, 2003.
[10] D. Boneh, G.D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public Key Encryption with Keyword Search,"
[11] Y.-C. Chang and M. Mitzenmacher, "Privacy Preserving Keyword Searches on Remote Encrypted Data,"
[12] R. Curtmola, J.A. Garay, S. Kamara, and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Con- structions," Proc. ACM Conf. Computer and Comm. [13] A. Singhal, "Modern Information Retrieval: A Brief Overview,"IEEE Data Eng. Bull., vol. 24, no. 4, pp. 35-43, 2001.
Security (CCS '06), 2006.
[14] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order- Preserving Symmetric Encryption," Proc. Int'l Conf. Advances in Cryptology (Eurocrypt '09), 2009.
[15] J. Zobel and A. Moffat, "Exploring the Similarity Space," SIGIR Forum, vol. 32, no. 1, pp. 18-34, 1998
[16] O. Goldreich and R. Ostrovsky, "Software Protection andSimulation on Oblivious Rams," J. ACM, vol. 43,1996
[17] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and Efficiently Searchable Encryption," Proc. Ann. Int'l Cryptology Conf. Advances in Cryptology (Crypto '07), 2007.
[18] S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, "Zerber+r: Top-k Retrieval from a Confidential Index," Proc. Int'l Conf. Extending Database Technology: Advances in Database Technology (EDBT '09),2009.
[19] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing," IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 5, pp. 847-859, May 2011.
[20] C. Wang, Q. Wang, K. Ren, and W. Lou, "Towards Secure and Dependable Storage Services in Cloud Computing,"

[21] C. Wang, S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy- Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, to appear.
[22] A. Swaminathan, Y. Mao, G.-M. Su, H. Gou, A.L. Varna, S. He, M.Wu, and D.W. Oard, "Confidentiality-Preserving Rank-Ordered
Search," Proc. Workshop Storage Security and Survivability, 2007.
[23] RFC "Request for Comments Database,"
http://www.ietf.org/rfc.html, 2012.
[24] B. Waters, D. Balfanz, G. Durfee, and D. Smetters, "Building an Encrypted and Searchable Audit Log," Proc. Ann. Network and Distributed Security Symp. (NDSS '04), 2004.
[25] F. Bao, R. Deng, X. Ding, and Y. Yang, "Private Query on Encrypted Data in Multi-User Settings," Proc. Int'l Conf. Informa- tion Security Practice and Experience (ISPEC '08), 2008.
[26] P. Golle, J. Staddon, and B.R. Waters, "Secure Conjunctive Keyword Search over Encrypted Data," Proc. Second Int'l Conf. Applied Cryptography and Network Security (ANCS '04),
[27] L. Ballard, S. Kamara, and F. Monrose, "Achieving Efficient Conjunctive Keyword Searches over Encrypted Data," Proc. Int'l Conf. Information and Comm. Security (ICICS '05), 2005.