International Journal of Security and Its Applications
Vol.8, No.3 (2014), pp.9-20
http://dx.doi.org/10.14257/ijsia.2014.8.3.02

Privacy- Preserving Keyword-based Semantic Search over Encrypted Cloud Data

Xingming Sun, Yanling Zhu, Zhihua Xia and Lihong Chen

Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science & Technology, Nanjing, 210044, China
School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing, 210044, China
sunnudt@163.com, zyl.112358@163.com, xia_zhihua@163.com, and *chen_81_lihong@163.com*

Abstract

To protect the privacy, sensitive information has to be encrypted before outsourcing to the cloud. Thus the effective data utilization becomes a significant challenge. Searchable encryption scheme has been developed to conduct retrieval over encrypted data. However, these schemes only support exact keyword search. Recent fuzzy search schemes mainly evaluate the similarity of keywords from the structure but the semantic relatedness is not considered. Our work focuses on realizing secure semantic search through query keyword semantic extension. Based on the co-occurrence probability of terms, the semantic relationship library is constructed to record the semantic similarity between keywords. We exploited architecture of two clouds, namely private cloud and public cloud. The search operation is divided into two steps. The first step expands the query keyword upon SRL stored in the private cloud. The second step uses the extended query keywords set to retrieve the index on public cloud. Finally the matched files are returned in order. Detailed security analysis shows that our solution is privacy-preserving and secure. Experimental evaluation demonstrates the efficiency and effectives of the scheme.

Keywords: *secure, rank, keyword-based, semantic similarity, encrypted cloud data*

1. Introduction

The Cloud provides convenient, on-demand high quality applications and services with a shared pool of configurable computing resources. Both companies and individuals can avoid capital expenditure by outsourcing their large volumes of data storage and computing work to the cloud in a pay-per-use manner[1]. However, Security risks prevent many users to take advantage of the cloud. Because they lose the control of their own data, and the privacy of their data is under the control of cloud service provider.

The straightforward solution to mitigate the concerns is to encrypt sensitive data before upload it to the cloud [1, 2].Nevertheless, data owners often want to share their data with other users. Typically, a user retrieves files of interest to him/her via keyword search instead of retrieving back all the encrypted files which may be impractical in cloud computing scenarios. Such keyword-based search technique has been widely used in our daily life e.g. Google plaintext keyword search. Although encryption of keywords can protect keyword privacy, it makes the traditional plaintext search techniques unsuitable in this scenario.

In recent years, searchable encryption (SE) techniques have been developed for secure outsourced data search. Traditional SE schemes provide the solution for securely search over encrypted data, but these techniques are not practically usable in context of cloud computing, since they only support exact keyword search [3-8]. For exact keyword search, the query keyword should exactly match the pre-set keywords from the files. To enhance the search flexibility and usability, some researchers have done some exploratory research on fuzzy keyword search [9-13]. However, these schemes mainly take the structure of terms into consideration to evaluate the similarity, and the terms semantically similar to keyword are not considered. Thus they result to low recall factor for the reason that many related files are omitted. Users may be able to input several additional keywords individually to complete a search. However, this approach greatly reduces the search flexibility and result in user searching experience frustrating. In addition, these fuzzy systems send back all relevant files solely upon presence/absence of the keyword, result-ranking still has not been considered.

In this paper, we design a practical encrypted search solution that support semantic search based on semantic relatedness. Semantic search reinforce the system usability by returning the exactly matched files and the files including the terms semantically similar to the query keyword. The co-occurrence of terms is used as the metric to evaluate the semantic distance between terms in semantic relationship library (SRL). In the proposed scheme, we exploit the architecture of two clouds, namely private cloud and public cloud. The private cloud performs the security-critical operations, while the public cloud performs the performance-critical operations. With the encrypted metadata set provided by the data owner, private cloud constructs the index and SRL. The semantic relationship library, which records the semantic similarity values of keywords, is stored in the private cloud for query extension. But the encrypted index is uploaded to the public cloud for efficient search. Thus the search operation is divided into two steps. The first step expands the query keyword upon SRL stored in the private cloud. The second step uses the extended query keywords set to retrieve the index on public cloud. Detailed security analysis shows that our solution is privacy-preserving and secure. Extensive experimental evaluation demonstrates the efficiency and effectives of the scheme.

The paper is organized as follows. The system model and threat model, design goals, notations, preliminaries and basic definitions are described in Section 2. Section 3 describes the basic framework of our scheme and Section 4 further gives the description of our semantic search scheme. The security analysis and performance evaluation are given in Section 5 and 6 respectively. In the end, Section 7 gives the conclusion of the paper.

2. Problem Definition

4.1. System Model

In this section, we describe the overall structure of our scheme. As illustrated in Figure his structure involves four different entities: data owner, data users, public cloud server, and trusted third party. In our paper, the third party could be a private cloud server.

Data owner has a collection of n text files $F = (F_1, F_2, F_3, \cdots, F_n)$. During the setup phase, the owner constructs a metadata for each file, then he outsourced the encrypted metadata set to the private cloud server. The text files are encrypted by using traditional symmetric encryption algorithm and uploaded to the public cloud server. Private cloud server constructs the inverted index and semantic relationship library using metadata set provided by data user. Then the Inverted index is outsourced to the public cloud server for retrieve.

During the search phrase, the authorized data users provide the search trapdoor to the private cloud server. In our paper, we assume the authorization between the data owner and users is appropriately done. Upon receiving the request, the private cloud server extends the query keyword upon SRL. Then he uploads the extended query keywords set to the public cloud. Upon receiving the search request, the public cloud retrieves the index, and returns the matching files to the user in order. Finally, the access control mechanism, which is out of the scope of this paper, is employed to manage the capability of the user to decrypt the received files.

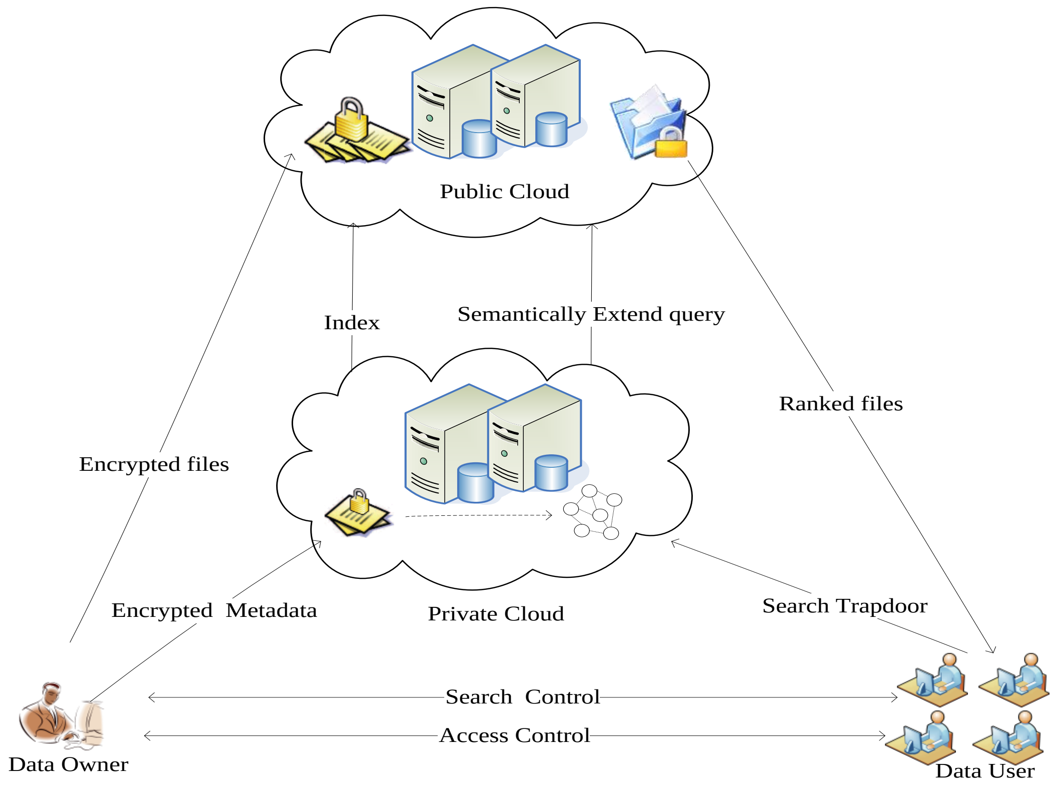

Figure 1. Architecture of the Semantic Search Scheme

4.2. Threat Model

In this paper, we consider an “honest-but-curious” public cloud server, which is described in previous searchable symmetric encryption (SSE) scheme [6, 7, 10, 11, 14, 15] Specifically, the cloud server honestly follows the designated protocol specification, but is “curious” to infer and analyze all data information available on the server to learn additional information. As for the private cloud server, we assume it is a trusted third part.

4.3. Design goals

Under the aforementioned model, we mainly want to solve the problem of secure ranked semantic search over cloud data. Our goals are summarized as follows.

(1) From a new perspective, we design a different fuzzy mechanism that support semantic search over encrypted cloud data by semantically extending the query keyword, and returns the ranked matched files in the end.

(2) Security guarantee: To prevent cloud server from learning the plaintext of the data files and keywords. Compared to the existing SSE schemes, the scheme should achieve the as-strong-as possible security strength.

(3) To achieve the above goals with minimum communication and computation overhead. Especially, the solution should reduce the overhead on data owner and users, who has limited storage and computing resource.

4.4. Notation

F −the plaintext file collection, denoted as a set of n data files $F = (F_1, F_2, \cdots, F_n)$.
C − the encrypted file collection, stored in the cloud server, denoted as $C = (C_1, C_2, \cdots, C_n)$.
$id(F_i)$ −the identifier of file F_i that can help uniquely locate the actual file.
W_i − the distinct keywords set extracted from F_i.
M −the encrypted metadata set, denoted as $\{M_1, M_2, \cdots, M_n\}$, where M_i is built for F_i.
EW −the encrypted distinct keywords set extracted from M, denoted as a set of m words $EW = (ew_1, ew_2, \cdots ew_m)$.
I −the inverted index built from the metadata set by the server, including a set of posting lists $\{I(ew_i)\}$
$F(w_i)$ − the set of identifiers of files that contain keyword w_i.
N_i −the number of files containing keyword w_i and $N_i = |F(w_i)|$.
T_w −the trapdoor generated for a query keyword w by a user.
S_w −the semantically extended keywords set of w, $S_w = (e_1, e_2, e_3 \cdots)$.

4.5. Preliminaries

(1) **Semantic Relationship**

Church *et al.*, had extended the word association in psycholinguistics which propose a statistical description of the semantic relationship between words through the co-occurrence of words [16]. To determine the similarity degree between terms, we use the data mining method to effectively find out the co-occurrence degree between terms in the collection. For two terms x and y, the mutual information I(x, y) is defined as

$$I(x, y) \equiv log_2 \frac{P(x, y)}{p(x)p(y)} \tag{1}$$

Here $P(x, y)$ is the probability of observing x and y together. $p(x)$ and $p(y)$ are the probabilities of observing x and y independently in the collection. The higher the semantic relationship between x and y is, the larger the mutual information $I(x, y)$ is.

Then normalize the mutual information into a value of relationship in interval [0, 1]. The semantic relationship library will be constructed as a weighted graph structure.

(2) **Inverted Index**

In information retrieval domain, inverted index is a widely used indexing structure which includes a list of mappings from keywords to the set of files that contain this keyword [17].To further rank the search results, the numerical relevance score is computed for each file. An example index structure of keyword w_i is shown in Tab.1. Here S_{ij} $(j = 1, \cdots, n_i)$ denotes the relevance score of file F_{ij} in response to w_i, n_i is the number of files contain keyword w_i.

Table 1. An Example of Inverted Index Structure

Keyword	w_i				
File ID	$id(F_{i1})$	$\mathrm{id}(F_{i2})$	$id(F_{i3})$	…	$id(F_{in_i})$
Relevance score	S_{i1}	S_{i2}	S_{i3}	…	S_{in_i}

4.6. Basic Definitions

(1) **File Metadata**

In our design, the file-metadata set is mainly used to construct the SRL. Meanwhile, we cloud also move the construction of index to the private cloud to reduce the computation burden on data owner. It is consisted of distinct words and the corresponding relevance score in the file. All file-metadata constitutes the metadata set of the dataset. An example of metadata set is shown in Table 2.

Table 2. An Example of Metadata Set

$\boldsymbol{id(F_1)}$	w_{11}	s_{11}	w_{12}	s_{12}	w_{13}	s_{13}	…
…	…	…	…	…	…	…	…
$\boldsymbol{id(F_n)}$	w_{n1}	s_{n1}	w_{n2}	s_{n2}	w_{n3}	s_{n3}	…

(2) **Ranking Function**

A ranking function is used to measure relevance scores of matching files to a given query in information retrieval. The most widely used measurement for evaluating relevance score is $TF \times IDF$ rule. TF (Term frequency) is used to measure the importance of the term within the particular file. IDF is used to measure the overall importance of the term within the whole collection. Thus the relevance score of single keyword can be computed using equation 2. It is widely used in many literature[18]:

$$Score(w, F_i) = \frac{1}{|F_i|} \cdot \left(1 + ln\, f_{i,w}\right) \cdot ln\left(1 + \frac{n}{f_w}\right) \quad (2)$$

Here w denotes the query keyword; $f_{i,w}$ is the TF of term w in file F_i; f_w denotes the number of files that contain keyword w. n is the number of files in the collection, while $|F_i|$ is the length of file F_i, obtained by counting the number of indexed terms in the file.

In our scheme, the matched files are ranked upon F_d's total relevance score. It is computed with equation 3.

$$TScore(w, F_d) = Score_w + \sum\nolimits_{e_i \in S_w} Score_{e_i} \quad (3)$$

Here $Score_w$ represents the relevance score of the input keyword; $Score_{e_i}$ represents the relevance score of extended keyword e_i.

3. The Ranked Semantic Keyword Search Scheme

To let the public server efficiently perform the result ranking meanwhile protect the privacy of the actual relevance score. To achieve more practical performance, we resort to the recent cryptographic primitive-order preserving encryption (OPE)[19]. Nevertheless, the original OPE is a deterministic encryption scheme, if not disposed properly, it will leak as much information as any deterministic encryption scheme does. In particular, the statistical information of the scores, such as the distribution slope, value range *etc.*, can be used to identify the specific keyword in the query[15]. Figure 4 shows an example of relevance score distribution for keyword "protocol". Therefore we modify the OPE to suit our requirement. A one-to-many OPE scheme is desired to reduce the amount of information leakage.

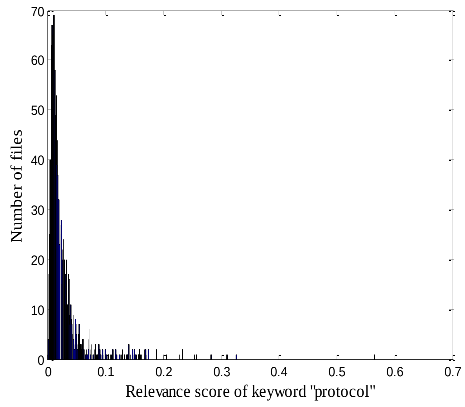

Figure 2. An Example of Relevance Score Distribution

4.1. One-to-Many Order-Preserving Encryption

In the modified encryption scheme, during the process of selecting the ciphertext, the unique file ID is introduced as an additional random seed. Thus the same plaintext will be mapped to a random value within the randomly assigned bucket in range R rather than the same ciphertext. Algorithm 1 shows the whole process, where $GetCoins(\cdot)$ is a random coin generator, $HYGEINV(\cdot)$ is the $HGD(\cdot)$ sampling function instance in MATLAB. The one-to-many OPE is denoted as $OM\text{-}OPE$ in the paper.

Algorithm 1one-to-many Order-preserving encryption

```
1:  procedure $OM\text{-}OPE_k(D,R,m,id(F))$
2:  while $|D|! = 1$ do
3:      $\{D,R\} \leftarrow BinarySearch(k,D,R,m)$;
4:  end while
5:  $coin \overset{R}{\leftarrow} GetCoins(k,(D,R,1||m,id(F)))$;
6:  $c \overset{coin}{\longleftarrow} R$;
7:  return $c$;
8:  end procedure

9:  procedure $BinarySearch(k,D,R,m)$
10: $M \leftarrow |D|; N \leftarrow |R|$;
11: $d = min(D) - 1; r = min(R) - 1$;
12: $y \leftarrow r + \lceil N/2 \rceil$;
13: $coin \overset{R}{\leftarrow} GetCoins\big(k,(D,R,0||y)\big)$;
14: $x \overset{R}{\leftarrow} d + HYGEINV(coin,M,N,y-r)$;
15: if $m \leq x$ then
16:     $D \leftarrow \{d+1,\cdots,x\}; R \leftarrow \{r+1,\cdots,y\}$;
17:   Else
18:     $D \leftarrow \{x+1,\cdots,d+M\}; R \leftarrow \{y+1,\cdots,r+N\}$;
19:   end if
20:   return $\{D,R\}$;
21: end procedure
```

The mapping scheme should be as random as possible to eliminate the predictability of the keyword specific score distribution. Obviously, the larger the size of range R is, the less specific characteristics will be preserved. However, considering the efficiency of HGD function, the size of range R cannot be unboundedly large. So the range size $|R|$

should be properly tradeoff between randomness and efficiency.

4.2. The Details of the Proposed Scheme

In this section, we present the detail of the proposed scheme. Our scheme consists of six algorithms ($KeyGen$, $BuildMD$, $BuildIndex$, $BuildSRL$, $TrapdoorGen$, $Search$).

- $KeyGen(k, l, \hat{l}, p)$: In this process, the data owner takes the security parameters $k, l, 1^{\hat{l}}, p$ as inputs. It generates random keys in this way: $x \overset{R}{\leftarrow} \{0,1\}^k$, $y \overset{R}{\leftarrow} \{0,1\}^l$ and outputs $Key = \{x, y, 1^l, 1^{\hat{l}}, 1^P\}$.
- $BuildMD(Key, F)$: In this process, the data owner builds the secure metadata for each file in file collection F.The details are given in Algorithm 2.

Algorithm 2 Build metadata set M

```
procedure $BuildMD(Key, F)$
for each file $F_i \in F$
    Extract the distinct words in $F_i$, denoted as $W_i = \{w_{i1}, w_{i2}, w_{i3} \cdots\}$;
    for $1 \le j \le |W_i|$
        1) calculate the score for keyword $w_{ij}$ according to equation 2, denoted as $s_{ij}$;
        2) compute $\pi_x(w_{ij})$ and $OM\text{-}OPM_y(s_{ij})$;
        3) store both as $\langle \pi_x(w_{ij}) || OM - OPM_y(s_{ij}) \rangle$ in metadata $M_i$;
    end for
    Insert $M_i$ to $M$
end for
return $M$
end procedure
```

- $BuildIndex(M)$: When receiving the secure metadata, the private cloud builds the inverted index. The details are given in Algorithm 2. Here ε denotes the maximum number of files containing word ew_i, *i.e.*, $\varepsilon = max_{0 \le i \le m} N_i$. The $\hat{l}$ padding $0'$s indicates the valid entry.

Algorithm 3 Build index I

```
procedure $BuildIndex(M)$
Initialization: extract the encrypted distinct words from $M$ as $EW = \{ew_1, ew_2, \cdots ew_m\}$;
for each $ew_i \in EW$
    Build a set of identifiers of files that contain $ew_i$, denoted as $F(ew_i)$;
    for $1 \le j \le |F(ew_i)|$
        1) extract the encrypted relevance score of $ew_i$ corresponding to the j-th file
           $F_j(ew_i)$, denoted as $es_{ij}$;
        2) store both with $\hat{l}$ padding $0'$s as $\langle 0^{\hat{l}} \| id(F_j(ew_i)) \| es_{ij} \rangle$ in the posting
           list of $I(ew_i)$;
        3) set remain $\varepsilon - N_i$ entries to random values of the same size as the existing
           $N_i$ entries of $I(ew_i)$;
    end for
    Insert $I(ew_i)$ into $I$
end for
return $I$
end procedure
```

- $BuildSRL(M)$: This algorithm is also run by the private cloud server to construct the semantic relationship library. It takes the metadata set M as inputs, and exploits datamining algorithm *e.g.,* Apriori algorithm, to discover the co-occurrence probabilities of keywords. Finally it uses equation 1 to quantify the semantic relationship values between keywords and stores the semantic relationship library SRL.
- $TrapdoorGen(Key, w)$: In this process, for a search input w, the user computes a trapdoor $T_w = \pi_x(w)$ and sends it to the private cloud. The second step is for the private cloud to extend the query $trapdoor$ and obtain the extensional query keywords set $T_w{}' = \{\pi_x(w), \pi_x(e_i)\}$, $e_i \in S_w$.
- $Search(T_w, I)$: Upon receiving the query request, the search phrase could be divided into two step: The first step is for the private cloud to extend the query trapdoor and sends the extensional query keywords set $T_w{}' = \{\pi_x(w), \pi_x(e_i)\}$, $e_i \in S_w$ to the public cloud. The second step is for the public cloud to locate the matching entries of the index via $\pi_x(w)$ and $\pi_x(e_i)$, which include the file identifiers and the associated order-preserved encrypted scores. The public cloud server then computes the total relevance score of each file to the query according to equation 3. In the end, the public server sends back the matched files in a ranked sequence, or sends top-k most relevant files.

4. Security Analysis

As mentioned in the thread model, the public cloud may use the available data to get additional information. Thus we mainly analyze the security on the public cloud.

4.1. Security Analysis for one-to-many OPE

The one-to-many order-preserving encryption introduce the file ID as the additional seed in the ciphertext chosen process, so the same plaintext will not be deterministically mapped to the same ciphertext c, but a random value in the allocated bucket form range Rby sampling, which help flatten the score distribution, and protect the keyword privacy form statistical attack.

However, if there are many duplicates of plaintext m, the ciphertext distribution may not be flattened effectively for the small size of assigned bucket in range R. So we should expand the range R properly to ensure the low duplicates on the ciphertext range, it will be difficult for the adversary to analyze which points in R belong to the same plaintext score.

4.2. Security Analysis for the Ranked Semantic Keyword Search

We estimate the security of the proposed scheme by proving the security guarantee stated in Section 2. That is, both the data files and the searched keywords are not leaked to the public cloud server.

- **File Confidentiality:**

The file is encrypted with traditional symmetric encryption algorithm. The encryption cipher is preserved by the data owner only, so the file confidentiality depends on the inherently security strength of the symmetric encryption scheme. Thus the file content is obviously protected.

- **Keyword Privacy:**

The index stored in the public cloud introduces some additional information compared to the original SSE, such as the encrypted relevance scores in the index. As previously discussed, if the data owner properly enlarges the range R, the relevance score will be randomly mapped to only a sequence of order-preserved numeric values with very low duplicates. The flattened encrypted relevance score distribution makes it difficult for the adversary to predict the plaintext score distribution, let alone predict the keywords.

5. Performance Analysis

To evaluate the performance of our proposed scheme, we implemented the secure search system using C++ on a windows machine with Intel Core 2 Duo CPU Processor running at 2.93GHZ, 2.94GHZ. The experimental evaluation was conducted on a real data set: Request for comments database(RFC)[20], this file set contains a large number of technical keywords. The overall performance evaluation of our scheme includes the cost of metadata construction, the time necessary for index and SRL construction as well as the efficiency of search.

(1) Metadata Construction

The main overheads for data owner are time cost and storage cost of metadata construction. To build a metadata for each document F_i in the dataset F, we should extract the distinct words and compute the associated relevance score, then encrypt the keywords and scores. The time cost of each entry directly depends on the number of keywords in the file, while the overall efficiency is also related to the number of the files in the collection. Table 3 lists the metadata construction performance. For the reason to eliminate the difference of various file set construction choices, both the metadata size and construction time listed are the average value.

Table 3. File Metadata Construction Overhead

Number of files	Per file metadata size	Per file metadata build time
1000	0.18KB	0.28s
2000	0.20KB	0.30s

(2) Index and SRL Construction

The index and SRL in our construction are built on private cloud server by scanning the whole metadata set. Figure 3 shows the time cost of building SRL against the increasing size of Mor dataset. Figure 4 shows that the whole index is nearly linear with the size of metadata set, namely the number of documents in the collection. In addition, taking into account the abundant computing resources on private cloud, the performance of building index and SRL is practically efficient.

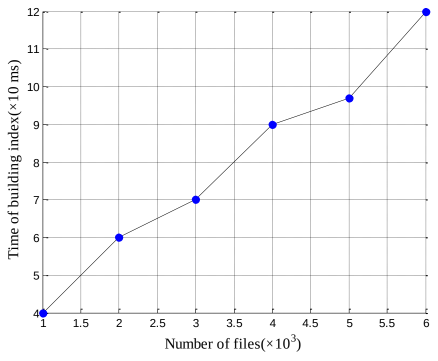

Figure 3. Time Cost for Building SRL

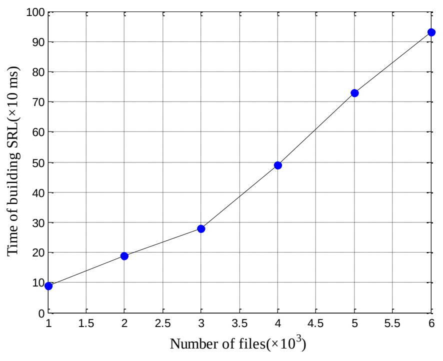

Figure 4. Time Cost for Building Index

(3) **Search Efficiency**

The search process includes query extension, fetching the posting list in the index, calculating the total relevance score and ranking the result in descending order. Compared to the original ranked search, our approach introduces the keyword extension cost, and the calculation cost of final relevance score. So the size of semantically extended keywords set is a factor to the query efficiency. Figure 5 shows the average time cost of query against the size of S_w. With result ranking, top-*k* search could return the most satisfied files more efficiently.

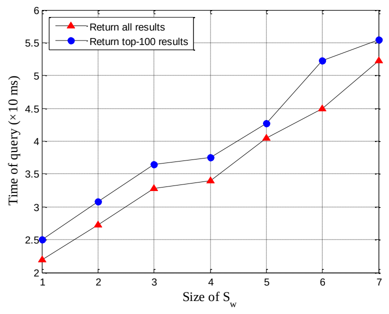

Figure 5. Time Cost of Query

(4) **Recall Factor of the Search**

By analyzing the search result, the overall recall rate is improved, and the query results are more in line with the user's actual intentions. *e.g.*, a user inputs a keyword 'protocol', the files which contain related words like 'internet', 'network' ,'authentication' will also be returned, in addition, the files which include most of the words will also be ranked forward.

6. Conclusion

As an initial attempt, we propose a secure semantic search scheme over encrypted cloud data. The proposed scheme could return not only the exactly matched files, but also the files including the terms semantically related to the query keyword. We use the co-occurrence probability of terms to get the semantic relationship of keywords in the dataset. It offers appropriate semantic distance between terms to accomplish the query keyword extension. To guarantee the security and efficiency, we exploited architecture of two clouds, namely private cloud and public cloud. SRL is stored in private cloud for security guarantee. The encrypted index is uploaded to the public cloud for search efficiency. We also derive a one-to-many OPE scheme to protect relevance score, while ensure the computing of total relevance score. Experimental evaluation demonstrates the efficiency and effectives of the scheme.

As our future work, we will design a new solution with only public cloud to solve the problem of semantic search over encrypted data. In addition, the semantic relationship value should be exploited to compute the total relevance score while protecting the privacy. Thus new crypto techniques still need to be designed to protect the semantic information while keep the ability to calculate the relevance score.

Acknowledgements

This work is supported by the NSFC (61232016, 61173141, 61173142, 61173136, 61103215, 61373132, 61373133), GYHY201206033, 201301030, 2013DFG12860, BC2013012 and PAPD fund.

References

[1] K. Ren, "Security challenges for the public cloud", Internet Computing, IEEE, vol. 16, **(2012)**, pp. 69-73.
[2] S. Kamara and K. Lauter, "Cryptographic cloud storage", Financial Cryptography and Data Security, ed: Springer, **(2010)**, pp. 136-149.
[3] D. X. Song, "Practical techniques for searches on encrypted data", Security and Privacy, S&P 2000. Proceedings. 2000 IEEE Symposium, **(2000)**, pp. 44-55.
[4] E.-J. Goh, "Secure indexes", Cryptology ePrint Archive, Report 2003/2162003.
[5] D. Boneh, "Public key encryption with keyword search", Advances in Cryptology-Eurocrypt 2004, **(2004)**, pp. 506-522.
[6] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data", Applied Cryptography and Network Security, **(2005)**, pp. 442-455.
[7] R. Curtmola, "Searchable symmetric encryption: improved definitions and efficient constructions", Proceedings of the 13th ACM conference on Computer and communications security, **(2006)**, pp. 79-88.
[8] M. Bellare, "Deterministic and efficiently searchable encryption", Advances in Cryptology-CRYPTO 2007, ed: Springer, **(2007)**, pp. 535-552.
[9] C. Wang, "Achieving usable and privacy-assured similarity search over outsourced cloud data", INFOCOM, 2012 Proceedings IEEE, **(2012)**, pp. 451-459.
[10] J. Li, "Fuzzy keyword search over encrypted data in cloud computing", INFOCOM, 2010 Proceedings IEEE, **(2010)**, pp. 1-5.
[11] M. Chuah and W. Hu, "Privacy-aware bedtree based solution for fuzzy multi-keyword search over encrypted data", Distributed Computing Systems Workshops (ICDCSW), 2011 31st International Conference, **(2011)**, pp. 273-281.
[12] C. Liu, "Fuzzy keyword search on encrypted cloud storage data with small index", Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference, **(2011)**, pp. 269-273.
[13] A. Ibrahim, "Approximate Keyword-based Search over Encrypted Cloud Data", e-Business Engineering (ICEBE), 2012 IEEE Ninth International Conference, **(2012)**, pp. 238-245.
[14] N. Cao, "Privacy-preserving multi-keyword ranked search over encrypted cloud data", INFOCOM, 2011 Proceedings IEEE, **(2011)**, pp. 829-837.
[15] C. Wang, "Secure ranked keyword search over encrypted cloud data", Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference, **(2010)**, pp. 253-262.
[16] K. W. Church and P. Hanks, "Word association norms, mutual information, and lexicography", Computational linguistics, vol. 16, **(1990)**, pp. 22-29.
[17] A. Singhal, "Modern information retrieval: A brief overview", IEEE Data Eng. Bull., vol. 24, **(2001)**, pp. 35-43.
[18] A. A. Moffat and T. C. Bell, "Managing gigabytes: compressing and indexing documents and images", Morgan Kaufmann, **(1999)**.
[19] A. Boldyreva, "Order-preserving symmetric encryption", Advances in Cryptology-EUROCRYPT 2009, ed: Springer, **(2009)**, pp. 224-241.
[20] Request For Comments Database [Online]. Available: http://www.ietf.org/rfc.html.

Authors

Xingming Sun Prof. Xingming Sun received his BS in mathematics from Hunan Normal University, China, in 1984, MS in computing science from Dalian University of Science and Technology, China, in 1988, and PhD in computing science from Fudan University, China, in 2001. He is currently a professor in School of Computer & Software, Nanjing University of Information Science & Technology, China. His research interests include network and information security, digital watermarking ,and cloud security.

Yanling Zhu She is currently pursuing her MS in computer science and technology at the College of Computer and Software, in Nanjing University of Information Science and Technology, China. Her research interests include information security and cloud security .

Zhihua Xia Dr. Zhihua Xia received his BE in Hunan City University, China, in 2006, PhD in computer science and technology from Hunan University, China, in 2011. He works as a lecturer in School of Computer & Software, Nanjing University of Information Science & Technology. His research interests include cloud security, and digital forensic.

Lihong Chen received his BE in Jiangsu University of Science and Technology, Zhenjiang, China, in 2011. She is currently pursuing her MS in computer science and technology at the School Of Computer and Software, Nanjing University of Information Science and Technology, China. Her research interests include cloud security, network and information security, digital watermarking.