Research on Ciphertext Retrieval based on Semantic Extension

Wenping Chen[a], Hongzhen Cui[b], Linlin Wang [c] and Fanqing Meng [d]

Zaozhuang University, Shandong 277160, China

[a]amo0114@163.com, [b] chz_367@163.com, [c]11266781@qq.com, [d]mengfanqing678@163.com

Keywords: ciphertext retrieval; keyword extension; multi-keyword ranking.

Abstract: In order to ensure data security and user privacy in the cloud, many companies and users store important data in the form of ciphertext. Therefore, how to efficiently and comprehensively search cloud ciphertext data without decryption has become the focus of research. In this paper, an effective and secure privacy protection scheme is proposed for outsourcing ciphertext data. The scheme uses a probabilistic public key encryption algorithm to encrypt data, and implements a multi-keyword sorting and keyword expansion search function in the retrieval process of encrypted data. The scheme uses the Apriori algorithm to expand the search keywords, and finds the ciphertext that is more in line with the user's needs through the expanded keyword set. Experiments show that this scheme can improve the speed of file encryption and decryption, document recall and accuracy, and reduce communication overhead

1. Introduction

Some organizations and experts at home and abroad have carried out a series of exploration in the field of data security and privacy protection in cloud environment, which naturally includes the research on the technology of searchable encryption. The scheme in [1] is used to solve the problem of dynamic updating of keyword and propose a new algorithm to generate trapdoor, and reduce the effect of the virtual word on the result ranking. In the literature [2], they propose a multi-key ciphertext retrieval scheme. They use the vector space model to represent document index and query requests. The two keyword weights are introduced into index and query vector in literature[2], Literature [4-7] have strengthened the index and specific search implementation for better protecting the privacy and searching documents. Literature [8] presents a vector space model and homomorphic encryption scheme to provide better search precision, however, homomorphic encryption increases the heavy computational burden and have the high communication overhead during the procedure of searching, the TRSE scheme cannot guarantee the recall of file. The key contributions of our work can be summarized as follows:

1. A multi-keyword Ranking search scheme based on keyword expansion is proposed. Accurate and comprehensive ciphertext search are supported by expanding search keywords.

2. Choosing an encryption algorithm that protects data privacy: use the probabilistic public key encryption algorithm [9] to encrypt the data and ease the computational overhead during encryption and decryption.

The rest of the paper is organized as follows: In Section 2, we explain system model used in our scheme. In Section 3, we describe specific search scheme. In Sections 4, we analyze experimental results. In Section 5, We summarize our work. The last part of the paper is reference.

2. Ciphertext Retrieval Scheme based on Keyword Extended (CRKE)

2.1 Scheme Overview

In order to better protect the privacy of outsourced data and reduce communication overhead, scheme uses probabilistic public key encryption and multi-keyword ranking search based on keyword extended (CRKE). Data owner uses the probabilistic public key encryption algorithm to

encrypt the outsourced data and upload it to the cloud server. After authorized users send search requests to the server, the cloud server uses the Apriori algorithm to extend the keyword semantics to retrieve the similar object files under the premise of not decrypting outsourced data, and then the top search results are returned to the authorized user for decryption. The scheme consists of two phases: preparation phase and retrieval phase.

2.2 Preparation Phase

In order to better protect the privacy of outsourcing data, we use probabilistic public key encryption algorithm to encrypt the document based on bilinear pairings which have less computational overhead and faster speed of encryption and decryption than homomorphic encryption algorithm.

• S_1, S_2, *r, s*, PK← keyGen (): The data owner chooses two random large prime numbers S_1, S_2 to calculate the product of two numbers access to its public key PK, using the extended Euclidean algorithm to obtain its private key *r, s*.

• C←dataEnc (F, *r*, PK): Convert the document to $f = \{m_1,..,m_n\}$,which m_i is a binary string of length h. Algorithm chooses a random number seed t to get *x*. and then get the pseudo-random bit p_i. And then its sequence and the document do XOR operation to get ciphertext C.

Encryption algorithm is as follows:

Preparation Phase

keyGen(): Select two large random primes S_1, S_2 ; PK = $S_1 \times S_2$

Calculate *r* and *s* using Euclidian algorithm where $rS_1+sS_2=1$;

Output public key is PK, private key is S_1,S_2,r,s.

buildIndex(F,K,*r, s*): Scan file collection F;

Extract keywords $k_1, k_2, \ldots, k_p$ from F_i ;

Record keywords' location: mi and paragraph number: L;

Construct keyword dictionary K with dummy Z;

For each $k \in K$, build F(k);

For each $k_i \in K$: for $1 \le j \le |F(k_i)|$:

fid = id($D_{i,j}$), //id(Di, j) is the j-th identifier in D(ki)

set $T[\pi_r (k_i \| j)] = fid$. //π is a pseudo-random permutation

for each $D_i \in D$: for each $k_j \in K$

z_j= TFIDF(k_j);

Output: I={T, **z**} .

dataEnc(F, *r*, PK): Let the file $f = \{m_1,..,m_n\}$ with length n, where each m_i is a binary string of length h and index $I(w_i)$;

select t as a seed ; $x = t^2 \bmod PK$;

for each i from 1 to n do // Generate the pseudorandom bits

$x_i = x_{i-1}^2 \bmod PK$;

$p_i = x \bmod 2$ //where p_i is least significant bits of x_i;

Calculate $C_i = p_i \oplus m_i$;

$I'(w_i) = p_i \oplus I(w_i)$

end for

$x_{n+1} = x_n^2 \bmod PK$. //generate next random

output: C = { C_1, C_2……C_n} .

2.3 Retrieval Phase

The way of searching keyword precisely matched ignores the retrieval target of the authorized user actually is Semantically related retrieval results, which is in line with people's Perceptual logic thinking. Cloud service providers receive the keywords from authorized users, and use the Apriori algorithm to extend semantics, and mining co-occurrence relation between word. We look at the document collection as a transaction T, and look at the keywords as the Itemset, we need to find all itemsets that satisfy the minimum support threshold. The concrete steps of generating frequent itemsets:

1. By scanning the keyword set, the support degree of each item is determined and get the frequent 1- itemsets Is_1={1-items};

2. The frequent 1- itemsets connected to generate 2- candidate set Ca_2= {(x, y)}, which x, y on behalf of keywords in keyword set.

3. Calculating the support degree of the candidate set, select the item that the support number is greater than the support threshold, and construct the k- frequent itemset.

Preparation Phase

ls_1={1-items} Find all the frequent 1-items
repeat
  k=k+1;
  Ca_2 =apriori-gen(Is_{k-1}) ; generate k- candidate set
  for each t∈T
    Ca_t = subset(Ca_k ,t); identify all candidates for transaction t
    for each candidate set Ca
      δ(Ca) ++ ; support degree increment
    end for
  end for
Extracting frequent k- itemsets ls_k .

Cloud service providers expand keywords by the Apriori algorithm to get the expansion keyword set S_w and query index. In order to avoid the number of extended keywords appear in document, we need to compare the comprehensive correlation CScore between extended keywords and documents. score represents keyword *w* and document relevance score; R_i represents the semantic relevance score of search keywords and keywords.

$$Cscore=score_w+\sum s_{i\in Sw}\, score_{si}\times R_i$$

CSP expand keywords to get the keywords set S_K to match the index after the trapdoor is sent to CSP and returns the top-k ciphertexts that are matched to the index to authorized user, otherwise, the ciphertext collection does not contain the files to be searched by the authorized user.

•Ω ←trapdoor (r, K ,$\overline{K}$): Authorized users use the pseudo-random permutation operation to generate t using the key from the data owner, and constitutes the query vector Ω with b.

•C←retrieval (Ω, I): The cloud server invokes the algorithm to use the vector value t in the trapoor to find the document set id by positioning the cloud index. For each encrypted document C_j, cloud server calculates the correlation score vector b in trapdoor and the value of the TF-IDF value in the index get similarity top-k sorted set C={C_1, C_2,…C_k}. Return it to the user to decrypt.

•M ←dataDec (S_1, S_2, r, s, PK, C): Data user receives the similarity ciphertext set C retrieved from the cloud server, and decrypting to get the plaintext set M= {f_1, f_2,…..f_k}.

Search Procedure is as follows:

ciphertext retrieval

trapdoor(r ,K, K): Tk=$\sum_{i=1}^{m} H(,K)$r
output: Tk
use Apriori algorithm expand the search keyword Tk to get the extended keyword set Sk ;
retrieval (Tk, I): for i from 1 to n
if(I'(wi)== S_k)
add matching ciphertext to the file list Ci;
t++;
end for
for i from 1 to t
Cscore= r_mw+$\sum$si∈Sw r_msi×Ri
end for
for i from 1 to k
get top-k of Ci
end for
output: C= {C1, C2,…Ck}
dataDec (S_1,S_2,r,s,PK, C):
Compute d_1= (S_1+1)/4)$^{n+1}$mod(S_1-1);
d_2= (S_2+1)/4)$^{n+1}$mod(S_2-1);
compute $p=x_{n+1}^{d1}$ mod S_1;
$q=x_{n+1}^{d2}$ mod S_2;
compute x=(prS_1+qsS_2) mod PK;
for i to n do
$x_i = x_{i-1}^2$ mod PK ;
p_i is the h least significant bit of x_i ;
end for
compute $f_i = p_i \oplus C_i$;
output: M ={f_1,f_2,…..f_k}.

3. Experiment

For data encryption, we use the probability of public key encryption algorithm, it is better than the existing scheme TRSE encryption algorithm, because it only needs a modulo can encrypt the h-bit plaintext, performance comparison is shown in Fig. 1. Fig. 2 shows the frequency of the selected random keywords such as "network" in the document set, that is each word frequency corresponds to how many documents for specific keywords. Fig. 3 shows the time consumption for returning the match file with query extension keyword set S_k when the document number is 2000 and the query keyword is 500. We can know that the number of extended query keywords has little effect on query time. Fig. 4 is the time to return the top-k file when the size of query extension keyword set S_k is 6 and the number of file collections is 1000. Fig. 5 shows the computation cost of authorized user for decrypting the file.

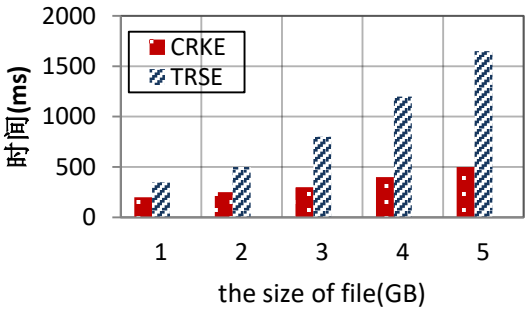

Fig. 1 The computation cost of encryption files

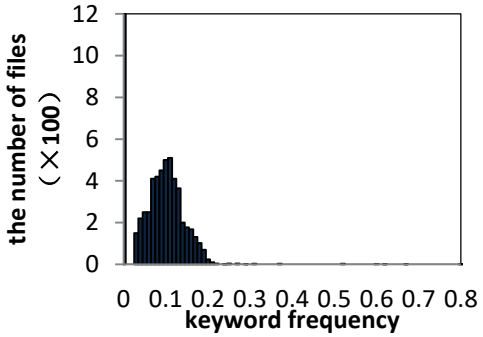

Fig. 2 The data distribution of the keyword 'network'

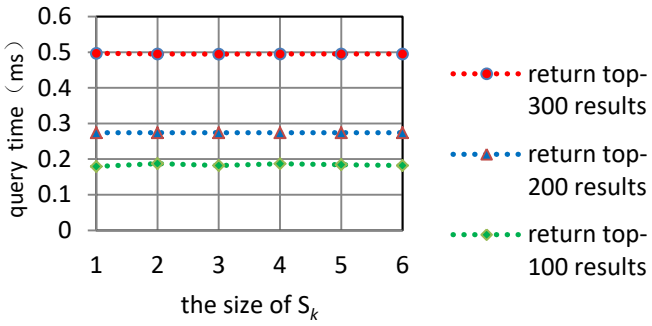

Fig. 3 the query time to return the match file with the different number S_k

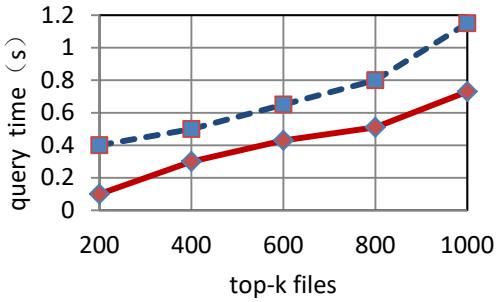

Fig. 4 The time to return the top-k file

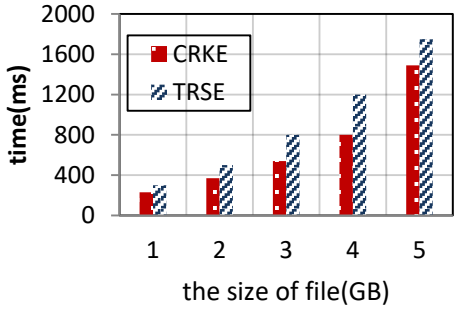

Fig. 5 The computation cost of decrypting the file

4. Summary

In our scheme, we use probabilistic public key encryption to encrypt the file collection and uploaded the encrypted documents and indexes to the cloud server. The authorized user uses the hash

collision function to generate the trapdoor and sends it to CSP searching the matching cipertext. CSP uses Apriori algorithm to expand keywords and search index to match the cipertext.

References

[1] Xu Z, Kang W, Li R, et al. Efficient Multi-Keyword Ranked Query on Encrypted Data in the Cloud[J].2012, 90(1):244-251.

[2] Orencik C, Kantarcioglu M, Savas E. A Practical and Secure Multi-keyword Search Method over Encrypted Cloud Data[C]// IEEE Sixth International Conference on Cloud Computing. 2013:390-397.

[3] Sun W, Wang B, Cao N, et al. Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking[J]. IEEE Transactions on Parallel & Distributed Systems, 2013, 25 (11):71-82.

[4] Cao N, Wang C, Li M, Ren K, Lou W. Privacy-preserving multi-keyword ranked search over encrypted cloud data. IEEE Trans Parallel Distrib Syst 2014;25 (1):222–33.

[5] Yang K, Zhang K, Jia X, et al. Privacy-Preserving Attribute-Keyword Based Data Publish-Subscribe Service on Cloud Platforms[J]. Information Sciences, 2016.

[6] Pasupuleti S K, Ramalingam S, Buyya R. An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing[J]. Journal of Network & Computer Applications, 2016, 64(C):12-22.

[7] Xia Z, Wang X, Sun X, et al. A Secure and Dynamic Multi-keyword Ranked Search Scheme over Encrypted Cloud Data[J]. IEEE Transactions on Parallel & Distributed Systems, 2015, 27(2):1-1.

[8] Yu J, Lu P, Zhu Y, et al. Toward Secure Multikey word Top-k Retrieval over Encrypted Cloud Data[J]. IEEE Transactions on Dependable & Secure Computing, 2013, 10(4):239-250.

[9] Tten, I.H.W. and T.C. Bell, managing giga bytes - compressing and indexing documents and images morgan kaufmann publ shers.

[10] Fan K, Wang X, Suto K, et al. Secure and Efficient Privacy-Preserving Ciphertext Retrieval in Connected Vehicular Cloud Computing[J]. IEEE Network, 2018, 32(3):52-57.