Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data by Chaos Based Arithmetic Coding and Confusion

Mengting Hu[1], Hang Gao[1], Tiegang Gao[2]

(Corresponding author: Tiegang Gao)

College of Computer and Control Engineering, Nankai University[1]
Tianjin Haihe Education Park, No. 38, Tongyan Road, Tianjin, China
College of Software, Nankai University[2]
Tianjin Haihe Education Park, No. 38, Tongyan Road, Tianjin, China
(Email: gaotiegang@nankai.edu.cn)

(Received June 8, 2017; revised and accepted Sept. 30, 2017)

Abstract

In this paper, a kind of secure and efficient ranked keyword search over outsourced cloud data by chaos based arithmetic coding and confusion is proposed. In the proposed algorithm, data owner firstly extracts keywords and generates index for files set for every keyword, in order to protect the sensitive score information relative to file, logistic map based arithmetic coding is used to give order-preserving mapping from original score to arithmetic coding, moreover, the number of relevant file to keyword is expanded and chaos based confusion algorithm is used to enhance the security of the algorithm. Secondly, for the authorized users, they hold different authorized key, and generate different trapdoor even for the same keyword, this is achieved by the idea of least significant bit replacement (LSBR). Upon receiving the trapdoor, cloud server first re-confuses to restore the order-preserved coded scores, and then identifies the associated files in a ranked sequence according to the coded scores. The proposed scheme can guarantee the security of the file, index and inquiry; make it impossible to disclose the relation between trapdoor and keyword. Experimental results and analysis are given to testify the security and efficiency of the proposed scheme.

Keywords: Least Significant Bit Replacement; Order-preserving Mapping; Ranked Keyword Search

1 Introduction

Cloud computing is an emerging computing mode where the data owner can be permitted to store their data into the cloud, by this kind of pattern of outsourcing the data into the cloud, some enterprises and individuals need not buy any storage devices with the demand of increased storage space, and they can also enjoy high-quality services from a shared pool of configurable computing resources [3,4,6,20]. This makes cloud computing becomes popular, and various information, including sensitive and important personal e-mails, location information, enterprise documents are being outsourced into the cloud [1,2].

Data privacy also becomes an important issue while cloud computing is increasingly prevalent. When people outsource some personal or enterprise data into the cloud, this information may be leaked to unauthorized users, or the hacked. Although cloud service providers (CSPs) have some data security measures such as firewalls and virtualization, however, these mechanism don't protect user's privacy from CSPs itself due to the cloud storage providers are not trusted [8,13–15,26].

The traditional approach of privacy preserving of sensitive data is to encrypt data before the data is outsourced into the cloud [5,16,25], but this may affects the data application for authorized user. In the meantime, some authorized user may only want to use some specific data files, so, people proposed keyword-based search method [12,17,21,23], it permits user to select relative files to the interested keyword, just as the method used in plaintext search scenarios. Furthermore, different from the keyword search in plaintext, people proposed searchable encryption schemes, which lets user search encrypted data through keyword search [17,23]. But these methods have some drawbacks, one is that the search results gives no any relevance of the files with the keyword, users only know that these encrypted files contain interested keyword. Another problem is that user need spend much time to enquiry the encrypted data which cloud gives back, so as to get the desirable file. Because user has no knowledge of which file is mostly interrelated to the keyword. Based on above considerations, the ranked keyword search (RKS) in the cloud data has been proposed. The mechanism can operate the encrypted data by returning the matching files with some keyword in a ranked order according to certain criteria [27, 29, 30]. Obviously, the RKS greatly enhanced the usability of data in the cloud. In order to avoid leaking lots of sensitive frequency information against the keyword privacy, RKS combined with some order preserving schemes are given to protect the relation between the keyword and file from leaking [18, 22].

In this paper, a kind of secure and efficient ranked keyword search over outsourced cloud data by chaos based arithmetic coding and confusion is proposed. In the proposed algorithm, data owner firstly extracts keywords and generates index for files set for every keyword, in order to protect the sensitive score information relative to file, logistic map is combined with bisection method code to generate order-preserving mapping from original score to arithmetic coding, moreover, in order to enhance the security of the coded scores, the number of relevant file to keyword is expanded and chaos based confusion algorithm is used to shuffle the scores. Secondly, for different authorized users, even for the same keyword, as they hold different authorized key, so they generate different trapdoor, this is achieved by the idea of least significant bit replacement (LSBR). Upon receiving the trapdoor, cloud server first re-confuses to restore the order-preserved coded scores, and then identifies the associated files in a ranked sequence according to the coded scores.

The highlights of our work can be summarized as follows:

1) In order to avoid leaking any information about keyword and its corresponding scores to the file set, chaos based arithmetic coding and confusion is used to hide the original keyword, meanwhile, the enlargement of number of scores corresponding to certain keyword also provides better privacy-preserving for keyword and its scores.

2) Un-linkability of trapdoor is realized by the inspiration of LSBR. This means that different user generates different trapdoor even for the same keyword query, thus it can avoid adversary deduce the relation between some trapdoor and someone keyword.

3) Some analyses on the efficiency, security and programmability are given to show that the proposed scheme can be easily implemented even in the resource constrained mobile devices, and the proposed algorithm has high efficiency for data owner and user.

2 Preliminaries

Some basic assumptions based on real application for outsourced data management are given in this section. Concerning the roles in the cloud service is data owner, data user and cloud server, as depicted in Figure 1.

Data owner: He has some set of files, he wants to outsource these files to the cloud server, moreover he wants to keep the files encrypted, and these files can be searched by a series of keyword. In order to protect the file from attacks, he hopes to create secure ranked searchable index from keyword and store them on the cloud server.

Authorized user: He hopes to get a series of files relevant to certain or some keywords submitted to cloud server, and the cloud server can give ranker files in a criteria, thus the authorized user can easily obtain the files he want.

Cloud server: It stores the files and keyword index, when it receives the request from the user, it can inquiry index and return the search results according to some ranked relevance criteria.

Application server: Application server is a component-based product that provides middleware services for security and state maintenance, along with data access and persistence. In our mode, it is a trusted program that handles all application operations between users and an organization's backend business applications or databases, and it can also be neglected in this model.

2.1 Design Goals

This paper aims at presenting a secure and searchable encryption scheme for outsourced data in the cloud, the scheme can prevent cloud server from learning some plaintext information or encrypted files information; moreover, the proposed scheme has better communication efficiency. More specifically, the goals of the paper are given in the follows.

1) Privacy goal: Privacy goals include three points, which is data privacy, index privacy and keyword and enquiry privacy. Data privacy means the data in the cloud is secure and anyone including CSPs can't obtain the plaintext of the data stored in the cloud. Index privacy demands that adversary can't obtain information stored in the cloud, including keywords, and scores relative to the keyword. Enquiry privacy demands that trapdoor generated by query keywords should leak no information about the keywords.

2) Search and retrieval efficiency: The proposed scheme should have lower time complexity of search time, and moreover, the retrieval efficiency and accuracy also meets the demand with the explosive growth of document size in big data scenario.

2.2 Notations

Some notations used in the paper are described in the following.

C: The file set to be outsourced, denoted as a set of n data files;

W: The distinct keywords extracted from file collection C, denoted as a set of m words $W = (w_1, w_2, \ldots, w_m)$;

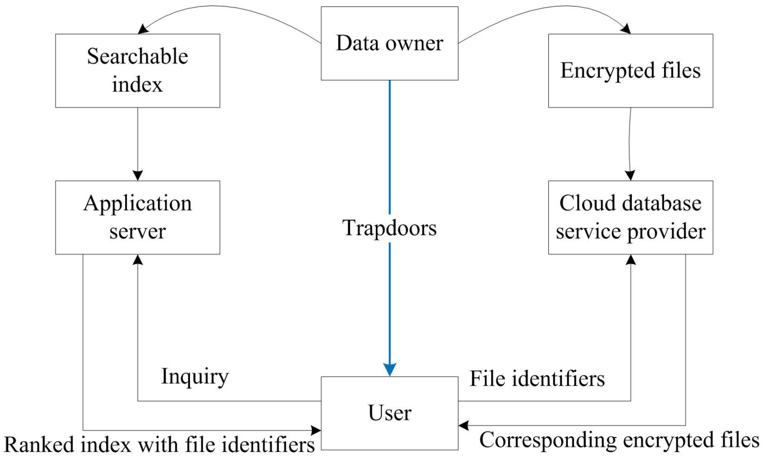


Figure 1: System model of cloud data management

$id(F_j)$**:** The identifier of file that can help uniquely locate the actual file;

$T(W_i, k_u)$**:** The trapdoor generated by a user as a search request of keyword W_i , k_u is the secret key of the user;

$\Gamma(W_i)$**:** The set of identifiers of the files in C that contain keyword W_i;

N_i**:** The amount of files containing the keyword W_i. Obviously, $N_i = \mid \Gamma(w_i)|$;

$Invertedindex$**:** inverted index is a list of mapping from keywords to the corresponding set of files that contain this keyword. In order to search the most related file to the keyword, ranking function is often used to achieve the goal.

In this paper, the ranking function is used to measure the relevance of files with certain keyword; it is often given in the form of relevance score. Without the loss of generality, here, the relevance score is selected as the following:

$$Score(Q, F_d = \sum_{t \in D} \frac{1}{\mid F_d|}(1 + \ln f_{d,t}) \ln(1 + \frac{N}{f_t}). \qquad (1)$$

where Q is the searched keyword; $f_{d,t}$ stands for the times of term t appears in the file F_d; f_t donates the file numbers that contains term t; N is the total number of files; and $|F_d|$ is the length of the file F_d. For more detailed description, one can see literature [4].

To realize fast search, the keywords, IDs of files, and the relevance scores are usually organized as an index structure named ”Inverted Index”. A typical example of Inverted Index is shown in Table 1. The cloud server can complete search task through comparing the relevance scores stored in the index which represent the importance level of each file for a certain keyword.

2.3 Logistic Map

Logistic map is a polynomial mapping; it is given in Equation (2)

$$x_{n+1} = rx_n(1 - x_n). \qquad (2)$$

For almost all initial conditions, the sequence of iteration is chaotic with the parameter $r = 4$, and it has been used in data shuffling and encryption for all kinds of application [9, 11].

2.4 Bisection Method Code

The bisection method in mathematics is a root-finding method that repeatedly bisects an interval and then selects a subinterval in which a root must lie for further processing. Here the method of data code based on bisection method is described in the following.

1) For a real number $x \in [0, 1)$, split the interval $[0, 1)$ into two segments $[0, \alpha)$ and $[\alpha, 1)$, then if $x \in [0, \alpha)$,we selected $[a_1, b_1) = [0, \alpha)$ and binary bit 0 is selected; else if $x \in [\alpha, 1)$, we select $[a_1, b_1) = [\alpha, 1)$ and the bit 1 is selected, where $\alpha \in (0, 1)$.

2) The new interval $[a_1, b_1)$ is split into two segments in the ration $\frac{\alpha}{1-\alpha}$. That is to say, $[a_1, b_1)$ is divided into $[a_1, \alpha \times (b_1 - a_1))$ and $[\alpha \times (b_1 - a_1), b_1)$. Similarly, one new bit 1 or 0 is produced, and new interval is selected. After k iterations, a k bit binary is produced, and an interval is generated.

Obviously, when $k \to \infty$, generated k-bit data is closely equal to x, so the binary code generated by bisection method is order-preserving, and it can be used to code real number.

3 The Proposed Scheme

In this section, the detailed description of the proposed algorithm is given, and some examples are also given to verify the effectiveness of the algorithm.

Table 1: Example of posting list of the inverted index

Keyword	w			
File ID	F_1	F_2	...	$F_{\Gamma(w)}$
Relevant score	6.2	1.3	...	7.6

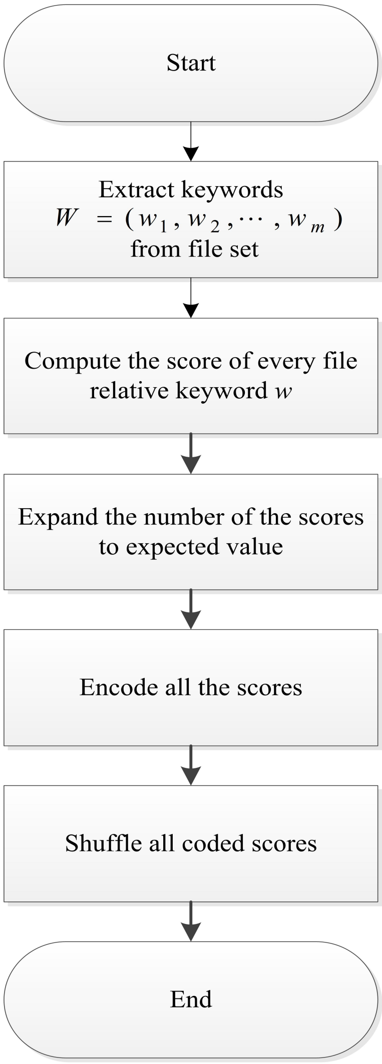

Start
Extract keywords $W = (w_1, w_2, \cdots, w_m)$ from file set
Compute the score of every file relative keyword w
Expand the number of the scores to expected value
Encode all the scores
Shuffle all coded scores
End

Figure 2: System model of cloud data management

3.1 Index Generation

Generation of index includes three steps, one is computation of relevance scores for every keyword; next is generation of binary code for all the scores, and the last one is the shuffling of coded scores to generate privacy-preserving index. The flowchart is depicted in Figure 2.

1) Computation of Relevance Scores:
   Firstly, data owner extracts the keywords $W = (w_1, w_2, \ldots, w_m)$ from the file set C, and then, the scores of relevant file for every keyword are calculated by Equation (1). Next, the order-preserving binary code of the scores will be given by chaos based bisection method.

2) Generation of Binary Code:
   For every keyword $w_i, i = 1, 2, \ldots, m$, the hash of the keyword is calculated, marked as H_i , then converts the hash value to initial value x_0 of logistic map which is expressed by Equation (2) using the same method as that of [10].

   Next, iterate the logistic map for $N_{insert} = N_{total} - N_i$ times to obtain N_{insert} random numbers, where N_{total} is the desired number of scores, and N_{insert} is the number of randomly inserted scores.

   Then, for all the scores related to keyword $w_i, i = 1, 2, \ldots, m$, labeled as $s_{i,t}, t = 1, 2, \ldots, N_{total}$, transforms them into the interval of $[0, 1)$, the interval is notated as $[a_j, b_j), j = 0$, the following step can be conducted to transform these scores into the binary code.

   a. Iterate the logistic map two times to obtain two numbers p and q, then, divide the $[a_j, b_j), j = 0$ into two sections according to the ration:

$$\alpha_j = \frac{\lambda_j}{\mu_j} \tag{3}$$

   where $\lambda_j = \frac{p}{p+q}$, $\mu_j = \frac{q}{p+q}$

   b. Obviously, the interval can be divided two sections, one is $[a_j, a_j + (b_j - a_j) \times \lambda_j)$, the other one is $[a_j + (b_j - a_j) \times \lambda_j, b_j)$, if the $s_{i,t} \in [a_j, a_j + (b_j - a_j) \times \lambda_j)$, we get binary bit ”1”, else the bit ”0” is given.

   c. If the length of binary code is equal to the desired length, then, all the bits construct the binary code of the score, else go to the Step 1) to continue to iterate until the length of the bits is enough.

Through the above step, all the binary code of the scores related to all the keyword could be obtained. The detailed flowchart of the procedure can be described in Figure 2.

3) Generation of Privacy-preserving Index:
In this procedure, in order to protect the binary code from attacks of adversary, we shuffle all binary code of the scores with respect to keyword. That is to say, for any keyword $w_i, i = 1, 2, \ldots, m$, the following steps are given to shuffle the binary.

   a. For above generated binary code of the scores, iterates the logistic map for N_{total} times to produce N_{total} numbers such as $x_i, i = 1, 2, \ldots, N_{total}$,and then rearrange these numbers in ascending order or descending order to form the sequences which may be expressed as $G_1 < G_2, \ldots < G_{N_{total}}$.
   b. Assume the position of x_i in $G_j, j = 1, 2, \ldots, N_{total}$ is L, $1 \leq L \leq N_{total}$, then, the binary code of $score_i$ which is in the position of i will be moved to the L^{th} position of the vector $s_{i,j}, j = 1, 2, \ldots, N_{total}$. Thus all the coded scores are totally permutated.

Obviously, the binary code of the scores will be totally confused through the above method, and no one can obtain any statistical information from the binary information, and for different keyword, the shuffling is different, this characteristic of dynamics can effectively protect the binary code from attacks.

After all the scores corresponding to certain keyword have been shuffled, data owner will store $I(w_i) = (id(F_{i,j})||s_{i,j})$ to the posting list in the cloud.

3.2 Retrieval Phase

In this phase, authorized user can retrieves ranked keyword search, and accordingly can get desired file, this procedure includes trapdoor generation and obtaining of ranked keyword index.

1) Generation of Trapdoor:
Trapdoor is used to encrypt the keyword, when authorized user wants to inquiry certain keyword, he sends it to the data owner, and the data owner will generate the trapdoor of the keyword and send it to the cloud server. Here, the trapdoor is an encrypted query with secret key $k_u$of certain user, and will be used for searching the file corresponding to keyword. It is denoted by $Trapdoor(w, k_u)$. The fulfillment of trapdoor function can be described as follows.

   a. For the keyword w, the 256-bit hash of the w is firstly calculated, then; convert the hash value into 32 bytes. In the meantime, for the 256-bit hash, we extract 2-bits (LSB) least significant of every byte, thus, 64-bit data is got, denoted by h_w^{64}.
   b. Calculate the hash of keyword and secret key k_u , denoted by h, and then convert it into 128-bit data by Equation (4), it is denoted by $T_w^{'} = h_{128}^{'} h_{127}^{'} \ldots h_1^{'}$.

$$h_i^{'} = h_i \oplus h_{i+128}, i = 1, 2, \ldots, 128. \quad (4)$$

   where $h = hash(w, k_u)$.
   c. For the secret key of inquiry k_u , convert it to the initial value of logistic map, and use Equation (2) to generate 64 different integers $p_1, p_2, \ldots, p_{64}$, which belong to $\{1, 2, \ldots, 128\}$, the $p_1, p_2, \ldots, p_{64}$ can be produced by Equation (5).

$$x_0 = mod((abs(x_0 - Floor(abs(x_0)) \times 10^{14}, 128) + 1. \quad (5)$$

   where $abs(x)$returns the absolute value of $x.Floor(x)$ returns the value of x to the nearest integers less than or equal to x, $mod(x, y)$ returns the remainder after division.
   d. Then, replace the corresponding bit value of position $p_1, p_2, \ldots, p_{64}$ in $T_w^{'}$ with the bit value of h_w^{64} in turn, thus, the new T_s of 128-bit data is generated.

Lastly, the trapdoor $T_w = (T_s||k_u)$ is generated, where T_s is a 128-bit binary data, and k_u is 32-bit binary secret key for the user.

2) Achievement of Ranked Keyword Index:
When the cloud server receives the trapdoor $T_w = (T_s||k_u)$ for an interested keyword w, the servers will inquiry the table of encrypted keyword, and obtain the file identifiers and the corresponding encrypted scores, and then return the ranked file according the binary code. The detailed steps can be given in the follows.

   a. The cloud servers firstly use the same method as the step 3) used in the generation of trapdoor to get 64 different integers $p_1, p_2, \ldots, p_{64}$, which belong to $\{1, 2, \ldots, 128\}$, then extracts the 64-bit data from T_s by the same method as that of 4) in the generation of trapdoor. The 64-bit data is labeled with T_c.
   b. Search the matched keyword. The cloud server searches encrypted keyword index, and transform the encrypted keyword into 32 bytes. And then extracts 2-bits (LSB) least significant of every byte, thus, a 64-bit data is got, denoted by h_s^{64}. If the h_s^{64} is equal to T_c, then the interested inquiry keyword is gotten.
   c. Get ranked file index. Use the same method as that in the procedure of generation privacy-preserving index to re-shuffle the coded scores corresponding keyword w to obtain original $s_{i,j}$.

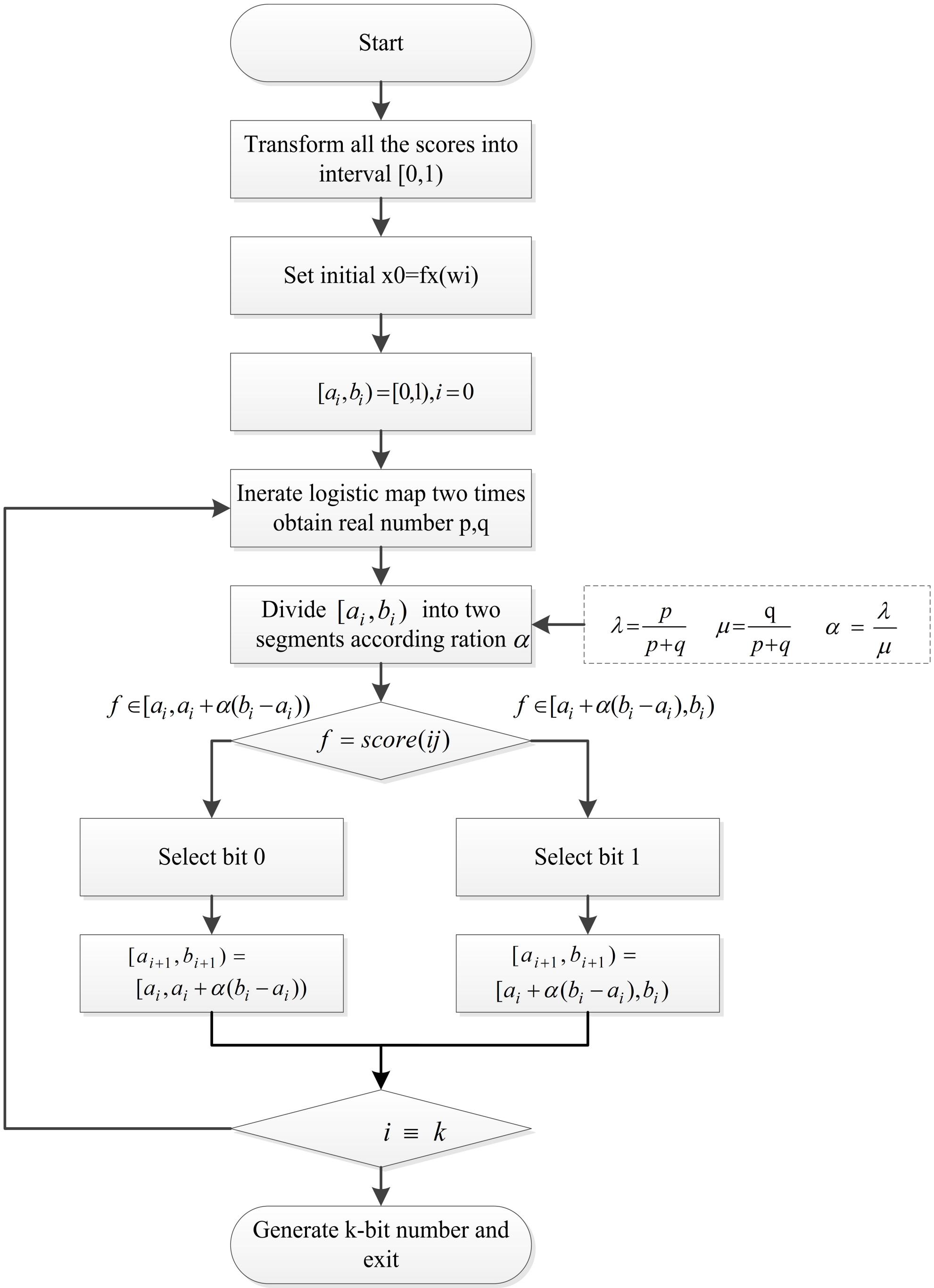

Start
Transform all the scores into interval [0,1)
Set initial x0=fx(wi)
$[a_i, b_i) = [0,1), i = 0$
Inerate logistic map two times obtain real number p,q
Divide $[a_i, b_i)$ into two segments according ration α
$\lambda = \frac{p}{p+q}$ $\mu = \frac{q}{p+q}$ $\alpha = \frac{\lambda}{\mu}$
$f \in [a_i, a_i + \alpha(b_i - a_i))$
$f \in [a_i + \alpha(b_i - a_i), b_i)$
$f = score(ij)$
Select bit 0
Select bit 1
$[a_{i+1}, b_{i+1}) = [a_i, a_i + \alpha(b_i - a_i))$
$[a_{i+1}, b_{i+1}) = [a_i + \alpha(b_i - a_i), b_i)$
$i \equiv k$
Generate k-bit number and exit

Figure 3: Generation of binary code

d. Obtain the front N_i encrypted scores corresponding keyword w, and discard the other inserted scores from the recovered N_{total} sequence of scores.

e. The server then selects the top-k most relevant files according to the coded scores and sends them to the users in the case that k is provided.

Remarks: The N_{total} stands for the desired number of score, this number may play an important role in the proposed scheme. In the basic SSE scheme [11], the number is $v = \sum_{i=1}^{m} N_i$. Here, it is recommended that $N_{total} \geq v$. In the meantime, if $k \leq N_i$, then server return back the top-k most relevant files, else server only return back valid file identifier.

Obviously, ranking keyword query in the proposed scheme has some kind of property of fuzzy query, that is to say, even for the same keyword inquiry, as different user has different secret key, cloud server may receive different trapdoor, but the different search may refer to the same keyword query, this may avoid adversary deduce the relation between trapdoor and keyword. Moreover, shuffled encoded scores also make statistical attacks impossible.

4 Experiments and Discussions

In this section, some experiments are given to testify the effectiveness of the proposed scheme, and some comparisons and analysis are also presented to show the performance and usability of the scheme. The experiments were done by Mathworks MATLAB version 12b in $IntelCpuP8400@2.26GHz, RAM3.00GB$. Here, assume that there are 10 files containing the keyword "digital watermark". The relevant scores of the file are list in the Table 2.

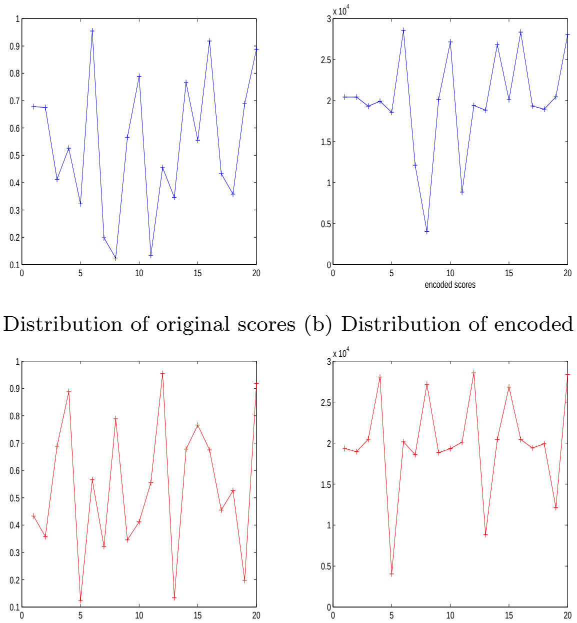
In order to resist the attack from the server, the file scores are expanded to 20, thus,10 random score values are given, such that 1.34, 4.55, 3.46, 7.66, 5.55, 9.18, 4.33, 3.58, 6.89, 8.88, and the generation of encrypted score is firstly given.

4.1 Experimental Results

Firstly, the hash of keyword is used to encrypt the keyword, and then we transform the hash value to the initial value of the logistic map, the next is to encode all the scores.

Here, the 256-bit hash value of keyword "digital watermark" and the 128-bit encrypted keyword are "F9437D7F3598D8FB1CD9EE8D1E27A1DAC7E4E96D3B8C56ABA3080B4A29ACCD80" and "36C3E9CF4032BE45E05CE65118F7E706", respectively.

After the relevant scores to the keyword digital watermark are coded, the distribution of original scores and coded scores can be seen in Figure 4. It can be seen that the coded scores are order-preserving. In order to resist the statistical attacks, the scores are shuffled, and the distribution of shuffled original scores and coded scores are depicted in Figures 4(c) and (d).



(a) Distribution of original scores (b) Distribution of encoded scores

(c) Distribution of shuffled scores (d) Distribution of shuffled encoded scores

Figure 4: Data distribution of original and coded score

4.2 Experiment Analysis

1) Security Analysis:
Firstly, for access pattern and search pattern, if the same keyword w_i is requested in query by different user, the query trapdoor submitted to the cloud server is different, and moreover, for different keywords in the same user query, the generations of trapdoor are independent.

Secondly, the encrypted scores are randomly distributed after they are coded and shuffled, thus the original distribution of scores is totally disrupted, and this means that the score distribution is secure from the viewpoint of statistics features.

Lastly, despite the coded scores has order-preserving property, the final coded scores in the cloud has no relevance between any two adjacent scores, it can be seen from the experimental results in the Figure 4(c). Thus, the relevance of scores is concealed, the data is secure.

In a word, in the proposed scheme, the coded scores are the only information that adversary can utilize; cloud server can only get sorted coded scores. Even if cloud server can learn partial information from the confusion process, as it is a one-time pad for different keyword, and different keys are used for different

Table 2: Example of posting list of the inverted index

Keyword	w									
File ID	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
Relevant score	6.78	7.23.	4.12	5.26	3.22	9.55	1.98	1.24	5.66	7.89

keywords and scores, thus, the keyword privacy can be well preserved in the scheme.

For example, user A with the 32-bit key "4053415001" and user B with the key "2872283436" submit the inquiry keyword "digital watermark", respectively. The trapdoor generated by data owner for user A is "7B756725317B019806E2368E7D4AB07B"; and however, the trapdoor generated for user B is "71EA8EC1B46B023F6A626531E0E7F0AB". Apparently, the trapdoor is different, but when they are submitted to the cloud server, they are transformed and point to the same keyword "digital watermark".

As for the random distribution of coded scores, it can be explained by another example. Assume that the keyword "digital watermark" and "information hiding" all have the same relevant scores to the files as that in the Table 1, then distribution of coded scores for keyword "digital watermark" and "information hiding" can be depicted in Figure 5. It can be seen from the Figure 5 that the coded score is different and the distribution of the coded score is also different despite the original scores are the same for two keywords, so it is impossible to get any information from coded scores stored in the cloud.

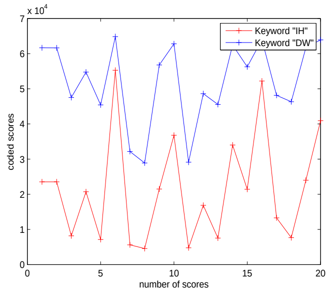


Figure 5: Different distribution of coded scores for the same original scores

2) Efficiency Measurement:

a. Index Construction In the proposed scheme, the length of the code for scores affects the performance of the algorithm, here, we tested the effect of the length of the code, shown in Figure 6, it can be seen from the Figure 6, the size of the score is 180, and the time cost for coding is about 35 milliseconds.

Because the 16-bit length of code is enough for representing a number, so the difference of efficiency affected by length of code is very small. As the code method is a simple binary operation, the algorithm efficiency is enough to meet the demands for cloud storage and computing, even for the resource constrained mobile devices.

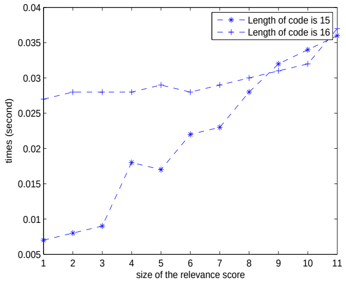


Figure 6: Efficiency of code for different length of code

b. Inquiry Efficiency

For the inquiry efficiency, some experiments are conducted. Firstly, the coded keywords are stored in cloud server, remote computer carries out inquiry of certain keyword, the test is given for the number of keyword be 2000,4000,6000,8000,10000,12000. The time efficiency can be depicted in Figure 7. It need to be explained that there are many factors, such as the deploy of the server and the design model of database all affect the inquiry efficiency, so it is more precise to give inquiry time in the server.

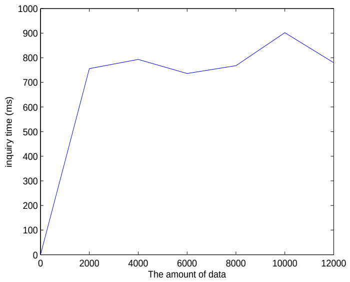


Figure 7: Inquiry time for different amount of data

c. Comparison

Firstly, from the security point of view, the proposed scheme can guarantee the security of data, index and inquiry, cloud server can't obtain any information relative some inquiry, and owing to the randomness of the coding, different keyword use different coding, therefore, it is also secure against attack of decryption. In this aspect, some existing scheme such as the nonlinear order-preserving index is insecure against attack [19].

Secondly, from the efficiency point of view, the proposed scheme generates binary code through bisection method; it is obvious that the algorithm has the higher efficiency of computation than that of the quasi-linear or nonlinear order-preserving coding, such as [19, 22–24].

Lastly, from the unlinkability point of view, different from some generation algorithms of trapdoor [17, 24], the proposed algorithm uses the idea of least significant bit replacement (LSBR) to fulfil the unlinkability of the trapdoor. The length of the trapdoor is 128-bit, the relative data to the keyword is randomly inserted into the 128-bit data, so it is difficult to deduce the relation between some trapdoors and some keyword.

As for the programmability, the proposed scheme can be easily implemented by any program language; it has the same better programmability as that of some existing scheme [19, 22].

5 Conclusions

In this paper, a kind of secure and efficient ranked keyword search over outsourced cloud data by chaos based arithmetic coding and confusion is proposed. In the proposed algorithm, data owner generates trapdoor of keyword and index for files set for every keyword, in order to protect the sensitive score information relative to file, logistic map based arithmetic coding is used to give order-preserving mapping from original score to arithmetic coding, moreover, the number of relevant file to keyword is expanded and chaos based confusion algorithm is used to enhance the security of the algorithm. For the authorized users, even they hold different authorized key, and generate different trapdoor, they can also enquiry the same interested keyword, and this is achieved by the idea of least significant bit replacement (LSBR). The detailed steps of flowchart of the proposed scheme are described in detail; some experiments are given to testify the usability of the algorithm. Lastly, some analysis and comparisons are given to highlight the merits of the proposed scheme.

In the future, the system model on the addition, deletion and modification of the files will be further researched, and the inquiry model of multi-keyword will be probed and analyzed.

Acknowledgments

The work was partly supported by the Program of National Science Fund of Tianjin, China (Grant NO. 16JCYBJC15700). The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] D. S. AbdElminaam, “Improving the security of cloud computing by building new hybrid cryptography algorithms,” *International Journal of Electronics and Information Engineering*, vol. 8, no. 1, pp. 40-48, 2018.

[2] M. H. R. Al-Shaikhly, H. M. El-Bakry, and A. A. Saleh, “Cloud security using Markov chain and genetic algorithm,” *International Journal of Electronics and Information Engineering*, vol. 8, no. 2, pp. 96-106, 2018.

[3] M. Armbrust, A. Fox, R. Griffith, *et al.* ”Above the clouds: A Berkeley view of cloud computing,” *Technical Report UCB-EECS-2009-28. Berkeley: University of California*, pp.1-23, 2009.

[4] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, I. Brandic, ”Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility,” *Future Generation Computer Systems*, vol. 25, pp. 599-616, 2009.

[5] Y. C. Chen, W. L. Wang, M. S. Hwang, “RFID authentication protocol for anti-counterfeiting and privacy protection”, in *The 9th International Conference on Advanced Communication Technology*, pp. 255–259, 2007.

[6] P. S. Chung, C. W. Liu, and M. S. Hwang, “A study of attribute-based proxy re-encryption scheme in cloud environments”, *International Journal of Network Security*, vol. 16, no. 1, pp. 1-13, 2014.

[7] R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, ”Searchable symmetric encryption: Improved definitions and efficient constructions”, in *Proceeding ACM Conferece Computer and Communications Security*, pp. 79-88, 2006.

[8] I. Damgard, T. Jakbosen, J. Nielsen, J. Pagter, ”Secure key management in the cloud,” *Cryptography and Coding*, pp. 270-289, 2013.

[9] T. Gao, Q. Gu, Z. Chen, ”Image encryption based on a new total shuffling algorithm”, *Chaos, Solitons, Fractals*, vol. 38, no. 1, pp. 213-220, 2008.

[10] H. Gao, M. Hu, T. Gao, R. Cheng, ”Double veriable lossless secret sharing based on hyper-chaos generated random grid”, *International Journal of Network Security*, vol. 19, no. 6, pp.1005-1015, 2017.

[11] Q. Gu, T. Gao, ”A novel reversible robust watermarking algorithm based on chaotic system,” *Digital Signal Processing*, vol. 23, no. 1, pp. 213-217, 2013.

[12] S. T. Hsu, C. C. Yang, and M. S. Hwang, “A study of public key encryption with keyword search”, *International Journal of Network Security*, vol. 15, no. 2, pp. 71-79, Mar. 2013.

[13] M. S. Hwang, C. C. Lee, T. H. Sun, "Data error locations reported by public auditing in cloud storage service," *Automated Software Engineering*, vol. 21, no. 3, pp. 373–390, Sep. 2014.

[14] M. S. Hwang, T. H. Sun, C. C. Lee, "Achieving dynamic data guarantee and data confidentiality of public auditing in cloud storage service," *Journal of Circuits, Systems, and Computers*, vol. 26, no. 5, 2017.

[15] T. Jaeger, J. Schiffman, "Outlook: Cloudy with a chance of security challenges and improvements," *IEEE Security Privacy*, vol. 8, no. 1, pp. 77-80, 2010.

[16] A. Khoshgozaran, C. Shahabi, "Private buddy search: Enabling private spatial queries in social networks", in *Proceedings of the IEEE International Conference on Computational Science and Engineering, Vancouver, Canada*, pp. 166-173, 2009.

[17] J. Li, Y. Lin, M. Wen,G. Yin, "Secure and verifiable multi-owner ranked-keyword search in cloud computing", in *Proceedings of International Conference on Wireless Algorithms, Systems, and Applications, Qufu, China*, pp. 325-334, 2015.

[18] K. Li, W. Zhang, C. Yang, N. Yu, "Security analysis on one-to-many order preserving encryption-based cloud data search", *IEEE Transactions on Information Forensics and Security*, vol 10, pp. 918-1926, 2015

[19] D. Liu, S. Wang, "Nonlinear order preserving index for encrypted databased query in service cloud environments", *Concurrency and Computation-Practice and Experience* vol. 2513, pp. 1967-84, 2013.

[20] L. Liu, W. Kong, Z. Cao, J. Wang, "Analysis of one certificateless encryption for secure data sharing in public clouds," *International Journal of Electronics and Information Engineering*, vol. 6, no. 2, pp. 110-115, 2017.

[21] Q. Liu, G. Wang, J. Wu, "Secure and efficient privacy preserving keywords searching for cloud services," *Journal of Network and Computer Applications*, vol. 35, pp. 927-933, 2012.

[22] Z. Liu, X. Chen, J. Yang, C. Jia, L. You, "New order preserving encryption model for outsourced databases in cloud environments", *Journal of Network and Computer Applications*, vol. 59, pp. 198-207, 2016

[23] Y. Lu, "Privacy-preserving logarithmic-time search on encrypted data in cloud," in *Proceedings of 19th NDSS, SanDiego, California, USA*, 2012. (http://dblp.uni-trier.de/db/conf/ndss/ndss2012.html#Lu12)

[24] S. K. Pasupuleti, S. Ramalingam, R. Buyya, "An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing," *Journal of Network and Computer Applications*, vol. 64, pp. 12-22, 2016.

[25] K. Puttaswamy, S.Wang, T. Steinbauer, D. Agrawal, A. Abbadi, C. Kruegel, B. Zhao, "Preserving location privacy in geosocial applications", *IEEE Transactions on Mobile Computing*, vol. 13, no. 1, pp. 159-173, 2014.

[26] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1-11, 2011.

[27] C. Wang, N. Cao, K. Ren, and W. Lou, "Enabling secure and efficient ranked keyword search over outsourced cloud data", *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 8, pp. 1467-1479, Aug. 2012.

[28] I. H. Witten, A. Moffat, T. C. Bell, "Managing gigabytes: Compressing and indexing documents and images", *IEEE Transactions on Information Theory*, vol. 41, no. 6, 1995.

[29] Z. Xu, W. Kang, R. Li, K. Yow, and C. Xu, "Efficient multi keyword ranked query on encrypted data in the cloud", in *Proceeding IEEE 19th International Conferece Parallel Distribution System*, pp. 244-251, 2012.

[30] W. Zhang, Y. Lin, S. Xiao, J. Wu, S. Zhou, "Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing", *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1566-1577, 2016.

Biography

Mengting Hu was born in Shanxi Province, China, in 1993. She received the B. S. degree in Software Engineering from Tongji University, Shanghai, China, in 2015. She is currently working toward the M. S. degree in Software Engineering from Nankai University, Tianjin, China. Her research interests include cloud computing and Information Retrieval.

Hang Gao was born in Tianjin City, China, in 1992. He received the B. S. degree in Software Engineering from University of Electronics Science and Technology of China, Chengdu, China, in 2015. He is currently working toward the M. S. degree in Software Engineering from Nankai University, Tianjin, China. His research interests include information security and cloud computing.

Tiegang Gao received Ph. D degree from Nankai University, Tianjin, China in 2005. He is a professor in college of software, Nankai University, China since 2006. His research interests include cloud computing and information security, he has published or co-authored more than 100 papers in related field.