Scientific Research Publishing

Journal of Computer and Communications, 2017, 5, 149-165
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Security Model for Preserving Privacy over Encrypted Cloud Computing

Jassim R. Mlgheit[1], Essam H. Houssein[2], Hala H. Zayed[1]

[1]Faculty of Computers and Informatics, Benha University, Benha, Egypt
[2]Faculty of Computers and Information, Minia University, Minia, Egypt
Email: jassimrahman88@fci.bu.edu.eg, essam.halim@mu.edu.eg, hala.zayed@fci.bu.edu.eg

How to cite this paper: Mlgheit, J.R., Houssein, E.H. and Zayed, H.H. (2017) Security Model for Preserving Privacy over Encrypted Cloud Computing. *Journal of Computer and Communications*, **5**, 149-165.
https://doi.org/10.4236/jcc.2017.56009

Received: March 6, 2017
Accepted: April 27, 2017
Published: April 30, 2017

Abstract

In our today's life, it is obvious that cloud computing is one of the new and most important innovations in the field of information technology which constitutes the ground for speeding up the development in great size storage of data as well as the processing and distribution of data on the largest scale. In other words, the most important interests of any data owner nowadays are related to all of the security as well as the privacy of data, especially in the case of outsourcing private data on a cloud server publicly which has not been one of the well-trusted and reliable domains. With the aim of avoiding any leakage or disclosure of information, we will encrypt any information important or confidential prior to being uploaded to the server and this may lead to an obstacle which encounters any attempt to support any efficient keyword query to be and ranked with matching results on such encrypted data. Recent researches conducted in this area have focused on a single keyword query with no proper ranking scheme in hand. In this paper, we will propose a new model called Secure Model for Preserving Privacy Over Encrypted Cloud Computing (SPEC) to improve the performance of cloud computing and to safeguard privacy of data in comparison to the results of previous researches in regard to accuracy, privacy, security, key generation, storage capacity as well as trapdoor, index generation, index encryption, index update, and finally files retrieval depending on access frequency.

Keywords

Cloud Computing, Multi-Keyword Query, Ranked Query, Trapdoor, Privacy Preserving, Encrypted Cloud Data, Top-K Query, Cloud Security

1. Introduction

We can refer to cloud computing as being a remarkable and outstanding IT innovation in today's life. It is mainly based on the fact that cloud can produce resources and services of computing since it is a modern and distinguished technique. The services provided by cloud computing are numerous and can be classified into the following: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) as well as Software-as-a-Service (SaaS) [1]. These services are also offered on the largest data canters scale such as Amazon, Google, and even Microsoft which draws the attention of several numbers of customers around the globe. As regards minor as well as medium size businesses, cloud computing relocation develops noteworthy and big savings from the economic perspective. As a matter of fact, cloud computing relocation is mainly dependent on the model of "pay per use" in respect with prices, as the payment by the user is based on his using or consuming the available resources [2]. And in spite of all the advantages offered by the cloud computing relocation, there are however some obstacles and problems which may occur such as inter-provider data portability problem, energy conservation problem, as well as security problem [3].

Several methods are mainly proposing to provide a precise protection of data privacy upon outsourcing storage on the Cloud Server Provider (CSP) [4]-[9] [11] [12] [13]. These methods use certain cryptographic mechanisms so as to enhance the policies used for access control. Hence, the users are allowed to get the keys for the items of data to which they have access. Theoretically, we have too many cryptographic techniques which can be used to achieve this objective [12].

Furthermore, the next important issue regarding the encryption as well as access control is known as key management as the mechanisms used for key management mechanisms may provide good access control to the data that are being outsourced on CSPs [4]. There are many methods suggested for exploiting the hierarchical together with other relationships taking place among several items of data so as to minimize the number of distributed keys and make key management more simple [4] [5] [7] [8] [11]. In other words, in the event of changing access control policies for these methods or approaches, the key distribution should be carried out once again in order to ascertain that authorized users are only allowed to access. Also, one only feature be easy to run on the computer of the user, *i.e.* cloud computing systems interface software. It must be as simple as the commonly used web browser, and the cloud network shall be responsible for the rest [12] [13].

In addition, there are several approaches which are suggested mainly so as to assess the security of cloud computing and therefore propose a "trusted the third party" with the aim of making sure that security considerations are complied with in any cloud computing environment [14] [15]. So as to confront the disclosure of data, a typical solution shall be private data encrypting prior to their upload onto the cloud server. Firstly, we must verify that any data are invisible for the external users as well as administrators of the cloud. Secondly, we have some strict limitations of processing on encrypted data. For instance, standard searching algorithms of plain text are no longer in use nowadays. So as to carry out a keyword-based query, we should decrypt all data set regardless of the tiny matching result set.

In this paper, we will propose a new scheme to improve the performance of cloud computing and to safeguard privacy of data in comparison to the results of previous researches in regard to accuracy, privacy, security, key generation, storage capacity as well as trapdoor, index generation, index encryption, index update, and finally files retrieval depending on access frequency.

The paper will be organized as follows. In Section 2, we will present the works related to ours. In Section 3, we will present the Problem formulation. Section 4 we will describe Precision and rank privacy. Section 5 we will evaluate our model through Performance analysis. Finally, Section 6 will be the Conclusion.

2. Related Work

This section will cover a detailed review of the related works which are referred for to formulate our proposed model.

Kamara, S., & Lauter, K. [23], have submitted their proposal of a conceivable architecture design to be used in a cryptographic cloud storage. As soon as the data are being prepared for storage onto the cloud, the owner of data will create certain indexes as well as encrypt such data using a specific scheme for symmetric encryption (e.g., AES) by means of a unique key. Therefore, the indexes are being encrypted by means of a scheme of searchable encryption to further encrypt such unique key by a scheme of attribute-based encryption following the proper policy. Eventually, all of the encrypted data as well as indexes are being encrypted in a manner which can be verified afterward by the data verifier to check if they are integral by means of a storage proof. As such, this identical strategy is also used in several types of research. In their turn, all of Fu, Z., *et al.* [24], have also proposed the application of a deterministic algorithm for encryption with the aim of keywords' encryption, as well as using stream ciphers so as to carry out security post-encrypt keywords. Furthermore, Han, F., *et al.* [25], gave their proposal of a new technique for transforming the Key-Policy Attribute-Based Encryption (KP-ABE) to be Attribute-Based Encryption instead using the feature of Keyword Search (ABEKS). In order to render transformation feasible, the researchers were keen on defining the feature of weak anonymity, which is known as attribute privacy that is also incurring slight computational transparency. The so-called cipher text-policy Attribute-Based Encryption (CP-ABE) is be used for the first while the aim of implementing a thorough control known as "priority access". In the next step, the fundamental or principal scheme "KP-ABE" applied for supporting encrypted data search facility. Nevertheless, the ABEKS is somewhat vulnerable to security violation as it is not providing the adequate or satisfactory feature of Access Control Aware Search (ACAS); however, it may leak such a volume of documents which includes the checked words. As well, it may be of less efficiency compared to such methods depending on the search based on the index. This would perform a complete decryption of documents so as to recover the documents that are requested;

while the methods of search based on the index are only decrypting documents' identifiers which include but not limited to such the keywords to be searched. As far as Chen, R., *et al.* [26], are concerned, Asymmetric Searchable Encryption is proposed in this respect in which they introduced Public Key Encryption with keyword Search (PEKS), that is dependent on the assumption of Bilinear Diffie -Hellman (BDH). The schemes of Asymmetric Searchable Encryption are suitable for to any situation in which the part which is searching onto data cloud differ from the generating one. The principal drawback in this regard is related to the weak security control.

According to Liu, Q., *et al.* [27], there is a newly proposed a scheme of Secure as well as Privacy Preserving Keyword Search (SPKS), that is especially allowing the Cloud Service Provider (CSP) to partake decipherment process, to retrieve files which contain certain keywords that are only specified by users, with the objective of lowering all of the computational as well as communication overhead to be decrypted for users, with one condition that user data are being preserved and their as well as user querying privacy are maintained. The thorough analysis of performance reveals that SPKS scheme is fit to be applied in any cloud environment. Furthermore, Shiba Sampat Kale, P., & Lahane, S. R. [28], have proposed a plain idea related to Multi-keyword Ranked Search over Encrypted cloud data (MRSE) which is mainly dependent on protected inner product computation in addition to effective similarity for coordinate matching. Then, there are two meaningfully significant improved schemes of MRSE so as to attain numerous strict privacy preconditions using two different threat models. Also, the assignment of anonymous ID is to be used by the user in order to maintain and secure the utmost security of data onto the cloud server. So as to improve the experience of data search and the search service in general, there should be more advanced extension using both schemes in order to support as much more as possible search semantics. As well, all of Xia, Z., *et al.* [29], have proposed a semantic multi-keyword ranked search scheme over the encrypted cloud data (MRSE) that is concurrently meeting some stringent privacy requirements. First of all, the researchers have used "Latent Semantic Analysis (LSA)" so as to show the existing relationship between all of the terms as well as documents. This relationship between terms is inevitably taken. In the second place, they scheme we use is employing secure "k-nearest neighbor (k-NN)" so as to achieve secure search functions. According to this proposed scheme, the exact matching files are not only returned, but also this feature extends to returning the files which include terms latent semantically related to query keyword.

Also, Madane, S. A., & Patil, B. M. [30], have discussed in detail the multiple-keyword ranked search over encrypted cloud data problem and also constructed various security settings which are required. Based on various concepts of multi-keyword, the researchers selected an efficient principle for coordinate matching. Also, they started in the first place to suggest a secure inner data computation feature. As well, the researchers could achieve efficient ranking result by means of k-nearest neighbor method that is used as well in order to help the server encrypt the document by RSA Algorithm and also convert encrypted document to be a Zip file having an activation code, then this activation code shall send to user a request for download. Finally, Barde, C. R., *et al.* [31], firstly executed a plain idea of Single Keyword Search over Encrypted Data as well as Multi-keyword Ranked Search over Encrypted cloud data (MRSE) which is mainly dependent on protected inner product computation in addition to the effective similarity of coordinate matching. In other words, several matches are being used with the aim of capturing the data documents relevance of in relation to the search query. Then, there are two meaningfully significant improved schemes of MRSE so as to attain numerous stringent privacy preconditions using two different threat models. Also, the assignment of anonymous ID is to be used by the user in order to maintain and secure the utmost security of the data onto the cloud server. So as to improve the experience of data search and the search service, there should be more advanced extension using both schemes in order to support as much more as possible search semantics.

3. Problem Formulation

In this research paper, we will propose a new (SPEC) model with the aim of improving the previously used models as well as results of the previous researches in this field, mainly document keyword collection which represents as index, encrypted index as well as secure index, multi-keyword query, trapdoor which is an encrypted version of a query. We will describe hereinafter the threat model, abbreviations as well as the proposed model architecture, and eventually the proposed model construction.

3.1. Threat Model

In our proposed model architecture, we consider that the cloud server is "honest-but-curious" which is typically adopted by most previous searchable encryption schemes. In other words, the cloud server is implementing honestly the protocol and then returns back the search results in a correct manner, however, it is curious as well to deduce some important information while performing the execution of the protocol. In the well-established cipher text, the encrypted dataset, encrypted search query and the searchable index are made are available to the cloud server [16]. We have two main parts, first model "Ciphertext Model" which supposes that the CSP can see encrypted files as well as indexes. Second "Background Model", in which CSP can collect intentionally queries' statistical data (trapdoors). Therefore, CSP is capable of calculating the file containing keyword [17].

3.2. Proposed Model Architecture

The entire architecture of cloud data service system which involves four entities, is shown in **Figure 1**. The data owner, data user, administration server and cloud server. A data owner has a data documents collection for outsourcing

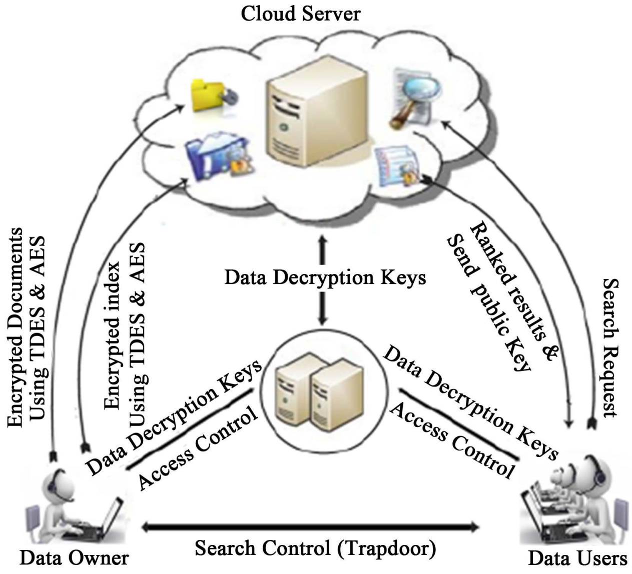


Figure 1. Architecture model of the search over encrypted cloud computing.

them to the cloud server in an encrypted form. So as to activate search capability over encrypted document collection for effective data utilization, the data owner before outsourcing, will be requested to build an encrypted searchable index from documents, he outsources to the index as well as the encrypted document collection to cloud server. For searching over the encrypted documents, the approved user will acquire a corresponding through search control mechanisms as he first obtains the trapdoor firstly, *i.e.*, the "encrypted" version of the search keyword, from the data owner, then submits the trapdoor to the cloud server. As soon as trapdoor is received from the data user, a cloud server is in charge of searching the index and returning the corresponding collection of encrypted documents. In order to improve document retrieval accuracy, search result must be ranked by cloud server in accordance with certain ranking criteria such as (coordinate matching). Furthermore, in if we would like to minimize communication cost, data user will send an optional number along with the trapdoor, and then, a cloud server is only sending back top-k documents which are relevant to the search query. At the end, access control mechanism used for managing decryption given to users as well as updating data collection by inserting new documents, updating existing ones, deleting existing ones.

3.3. Proposed Model Construction

In this section, we will present a detailed description of our proposed model which will be based on the following logarithms:

3.3.1. Key Generation

Our proposal is based mainly upon the following: a key generator which is able to generate new keys depending on three part (M1, M2, and S) and a set of operations. Results are merged to check the size of output equal the size of the plain text. In case the size of the output is equal to the size of plain text, therefore, merged key can be entered as a secret key. Algorithm 1. Illustrates the steps of the key generation proposal.

Algorithm 1. Secret Key generation

```
Procedure: Secret Key generation
The secret key consists of three parts.
    Converting first part to square matrix A.
        Making transpose to matrix A and stored in a
        Matrix B.
            Multiplying the A to B.
    Storing result in $M_1$
    Converting second part to square matrix C.
        Making transpose to matrix C and store in
        Matrix D.
            Multiplying the C to D.
    Storing result in $M_2$
    Making reverse to the third part and store in E.
        Making Xor operation between the third part
    And E.
        Storing result in S.
    Merging the three-part to present the new key in
    The following form:
    S connect $M_1$ connect $M_2$
    The new key possible to enter the master key.
End
```

3.3.2. Build Index

Having generated the key pairs, then the owner will build a file collection index. In general, we will utilize Kuzu, M., *et al.* [18] scheme to be the basis on which our index is built. Then, building the index will be shown in detail hereinafter in Algorithm 2.

Algorithm 2. Build Index

```
Procedure: Build Index
$K_{id}$ = Keygen ( $\psi$ )
For all Di $\in$ D do
    $F_i$ = extract features of Di
        for all $f_{ij} \in F_i$ and $g_k \in g$ do
            If $g_k(\dot{f}_{ij}) \notin$ bucket identifier list then
                Add $g_k(\dot{f}_{ij})$ to the bucket identifier list
            end if
        end for
    end for
    for all $B_k \in$ bucket identifier list do
        $Y_{Bk} = EncK_{id}(B_k)$
        add $Y_{Bk}$ to I
```

```
  end for
  return I
```

where D is data document collection; g is composite hash functions; ψ is security parameter.

3.3.3. Encryption Index

Having built the index, the owner will encrypt index so as to ensure the privacy of index. So far as there is limited computing power on data owner's part, we will have encrypts index [19]. Our process of encryption process will be detailed hereinafter in Algorithm 3.

Algorithm 3. Encryption index

```
Procedure: encryption index
Split the index I into two vectors {I', I"}
  for each element ij $\in$ I
    set I'j = I"j = Ij if sj $\in$ S is 1
    Otherwise I'j = 1/2 Ij + r, I"j = 1/2 Ij − r
Encrypt {I', I"} with (M1, M2) into $\{M_1^{t} \cdot I', M_2^{t} \cdot I''\}$
Output Enc SK (I) = $\{M1^{t} \cdot I', M2^{t} \cdot I''\}$ as the secure index.
```

Where r is a random number

3.3.4. Encryption and Decryption of Data

Having uploaded the data onto the cloud server, the owner will encrypt file collection so as to make sure of the privacy files. Meanwhile, the data owner is only having a limited computing power, and we have used *Tripple Data Encryption Standard* (TDES) and Advanced Encryption Standard (AES-128) Algorithms in order to encrypt files with the aim of ascertaining data privacy. Having received the matched documents from CSP for the purposes of search request corresponding, the approved user will decrypt such files by means of the private key as well as receive plain text using Tripple DES algorithm in order to decrypt the document. TDES takes a 64-bit long plaintext data block 56-bit input and in the meantime will generate a 64-bit block output. We conduct the same DES algorithm for three times on each of the data blocks. It is often extending DES key size of as per the algorithm to be applied three times successively by means of 3 different keys [20] [21]. Prior to the use of 3TDES, the user will generate and distribute a 3TDES key K firstly, that is composed of three main DES keys K_1, K_2 and K_3. It means however that actual 3TDES key has length 3 × 56 = 168 bits. The process of encryption will be detailed hereinafter in two Algorithms

Algorithm 4.1. Tripple Des of Encryption and Decryption

We can describe the encryption and decryption process as follows:-

1. Procedure: Encryption and Decryption
2. Encrypt the documents or files using single DES with first Key that is called K1.
3. Then, decrypt the output from Encrypt process with first key using single DES With second key that is called K2.
4. Finally, encrypt the output from decrypt process with second key using

Single DES with third key that is called K3.

5. The output from encrypt process with third key is encrypted document.

Decryption process of encrypted documents or files is a reverse process. The user will firstly, decrypt of documents or files using third key that is called K3, then encrypt it using second key that is called K2, and finally, decrypt of documents or files using first key that is called K1.

Algorithm 4.2. AES-128bits of Encryption and Decryption

We describe in detail the rounds of Advanced Encryption Standard (AES-128) Algorithms in order to encrypt files or documents and each round consist of four sub-processes as follows

1. Procedure: Encryption and Decryption
2. SubBytes:

Byte Substitution is a nonlinear byte substitution that operates independently on each byte by looking up on a fixed table is called (S-box) and the result of the 16 input bytes is a matrix of as well as four columns as well as four rows.

3. ShiftRows:

is shifted to the left for each the four rows of the matrix and transformation operates on the rows , it cyclically shifts on bytes in each row and bytes that fall off of the row are reinserted on the right side of the same row. The shift is carried out as follows:

- unchanged of The first row.
- shifted one byte (one positions) to the left of The second row.
- shifted two byte (two positions) to the left of The third row.
- shifted three byte (three positions) to the left of The fourth row.
- the output is a novel matrix containing the same byte but shifted.

4. MixColumns:

MixColumns is transformed process of all the columns that containing four bytes by means of certain mathematical function. that Such a function takes for each column four bytes as input and the result is four totally novel bytes, and novel bytes is replace with original bytes ,The result is a novel matrix containing 16 new bytes. This step is not executed in the final round.

5. Add round key:

In this Add Round Key operation which executed (XOR) operation to the 128 bits of the round key. If this is the final round then the result is encrypted documents or files. Executed XOR operations on results from mix column and round keys. For AES 128,128 bit XOR operations are executed. Otherwise, the resulting 128 bits are interpreted as 16 bytes and then we begin another similar round.

3.3.5. Trapdoor Generation

Having sorted the data in the cloud, if authorized user is looking forward to retrieving any file containing some keywords, he will compute the trapdoor (Twi) for keywords wi ∈ w and then resend it to the cloud server provider (CSP) in the form of search request [19]. The process of computing trapdoor will be detailed hereinafter in Algorithm 5.

Algorithm 5. Trapdoor generation

1. Procedure: Trapdoor
2. Compute Trapdoor
3. Send trapdoor to the (CSP)
4. The user gets the trapdoor information from the data Owner
5. For inserted keywords, the user computes the trapdoor
6. Then, the user sends trapdoor (Twi, k) to the CSP

Where k is an optional value, Twi is a compute trapdoor.

3.3.6. Ranked Search

Data user will send the trapdoor to the CSP. Having the information, first of all, the CSP will determine the files that may be accessed by data consumer (DC), and he will compute afterward matching score of each authorized file in the encrypted index set. Then the CSP will sort results depending on scores and will return back the top k files in the resulting set to the DC. In our trapdoor algorithm, whenever keyword access frequency is regarded, values of locations in the query vector are very likely to be determined by their corresponding access frequency. The process of ranked search will be detailed hereinafter in Algorithm 6.

Algorithm 6. Ranked search

1. Procedure: Ranked Search
2. for every level i from 1 to n do
3. If (I' (wi)== Twi)
4. Rank (Ri)/higher level which is matching query
5. end if
6. end for
7. Cloud server ranks matched documents according to rank.
8. Then, it sends top-k documents that have most relevant to the user R = {r1, r2 … rk} Where R is relevant documents, Twi is Trapdoor.

4. Precision and Ranked Privacy

The definitions of commonly used performance metrics will be used herein, which is precise to measure search result in an accurate manner as well as the rank privacy which can count the information leakage of search results. In order to assess the effect on search result accuracy, we use the definition "precision" [22]. In other words, "precision" of a top-k search will be defined as Pk = Ḱ/k whereas Ḱ represents the number the real top-k documents which are returned by cloud server. Meanwhile, we will assess the "rank privacy". The definition "rank privacy" is also used [22], *i.e.*, the rank privacy at point k is calculated as $P = \sum_{i=1}^{k} i / K^2$ Ṕ. For every document in the returned top-k documents, we will define the rank perturbation as Ṕi $= |ci' - ci|$ where ci′ is the ranking of document di in the retrieved top-k documents, and it is set to k if greater than k, ci is the actual ranking of document di in the data set and k denotes the number of top-k retrieved documents.

5. Performance Analysis

We have executed our own schemes using a laptop or a PC provided with Intel Core i5 processor of 3.3 GHz capacity as well as 4 GB RAM memory. The total number of simulation code is 5637 lines written in java (JDK) language with Sql yog, NetBeans IDE 7.1.2, Mozilla Firefox, using simple File *Transfer Protocol* (FTP) protocols and apache-tomcat server of similar computation power using Amazon EC2 M1 Medium instance. So as to generate ranked keyword query, we use on a random basis, a letter from a certain keyword, then it will be replaced with another one. We will allow mostly two ranked query of keywords. In the next section, we will give a detailed description charts which show our research results.

5.1. Generation Time and Storage Space of Secret Key

On the superior features of SPEC in comparison to previous solutions is that it can naturally extend keyword dictionary set at the minimal cost. In our experiments, we will firstly compare time consumption to generate secret keys of proposed Extended-Keygen algorithm with MKQE and MRSE algorithm when new keywords are introduced in the dictionary. Then, we compare storage consumption performance.

We can observe from Figure 2(a) that SPEC is more efficient than MKQE and MRSE. Basically, whenever original dictionary size is up to 1000, MRSE,

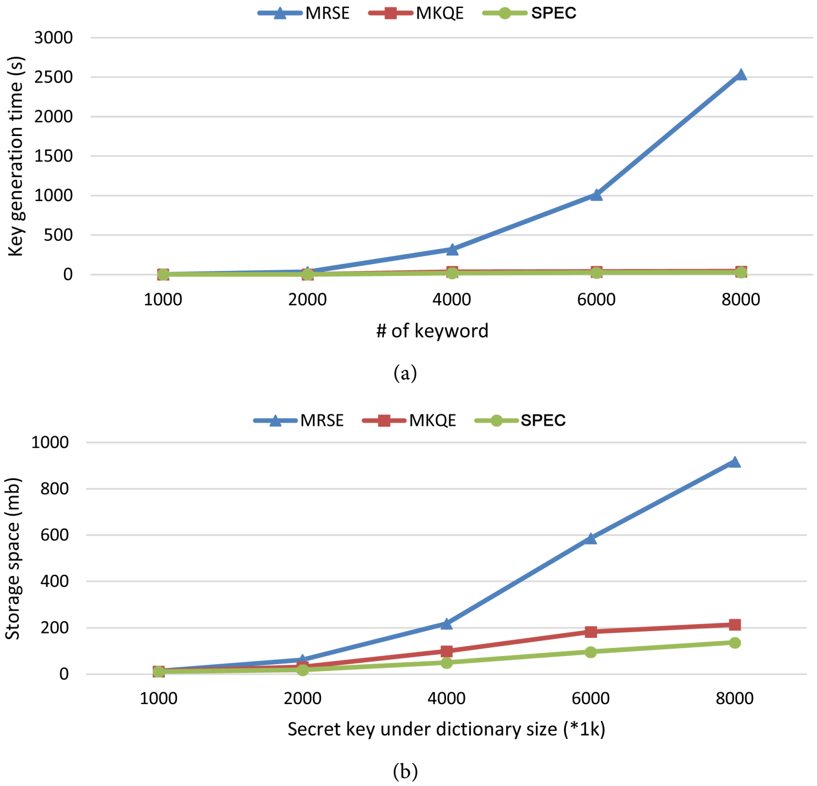


Figure 2. (a) Secret key generation overhead (s), starting from 1000 keywords; (b) the storage comparison under the same dictionary size.

MKQE as well as SPEC will use the same algorithm in order to generate secret keys, thus their performances are the same. The overhead for secret key generations in MRSE is gradually increasing compared to MKQE and SPEC. SPEC is better than MKQE and MRSE and the difference is minimal. Thus, the time consumption in MRSE is higher than MKQE and SPEC. Furthermore, the performance gap becomes even wider as more and more keywords are added. Apparently, SPEC has a better performance than MRSE and MKQE since it reuses an original set of indexes during keyword expansion. A number of elements required to be produced in matrices is too much smaller than MKQE as well as MRSE respectively.

Also, we compare storage consumption to update keyword dictionary as well as other data structures in our scheme with MKQE and MRSE. Result in Figure 2(b) indicates that SPEC consumes less space. As the size of the dictionary increases, SPEC saves, even more, storage spaces than MKQE and MRSE. The reason is that in SPEC, we use partitioned matrices and a great quantity of unused elements which are not stored. Due to the help of linked matrix list, SPEC can make sure that the space consumption grows linearly according to the expansion of dictionary.

5.2. Time of Trapdoor and Index

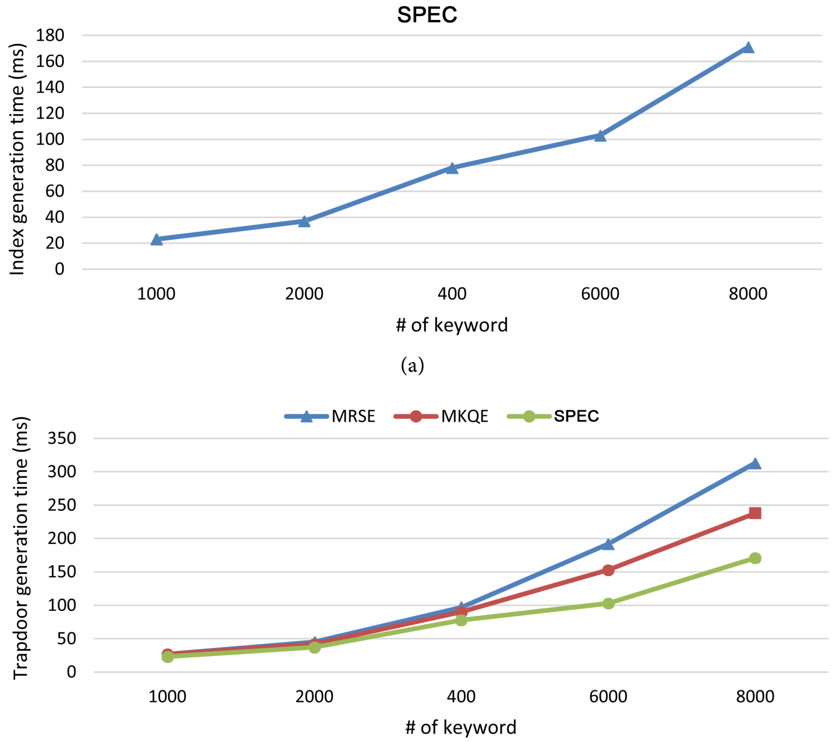
In order to evaluate the SPEC performance, we are going to analyze as well time consumption in regard to various operations. In our experiments, we will assess the index construction time, trapdoor construction time, index update time and the finally Index encryption time.

In Figure 3(a) we notice index generation time for the index, the generation time which is increasing in a linear manner with respect to a number of inserted keywords. Trapdoor generation time is almost the same as index generation time because of the identical procedure.

In Figure 3(b) a comparison is held between trapdoor generation time consumption of MRSE, MKQE on the one hand and SPEC on the other hand. As per results, we can observe that it shows in all scenarios, SPEC outperforms MKQE and MRSE, and therefore performance gap is larger with the increase in keyword dictionary size.

As per Figure 4(a) results of time consumption metric as soon as the dictionary is extended. In this set of experiments, we compare time spending on index update operation in SPEC with the time to MKQE the complete set of file indexes as well MRSE since the dictionary expands. Also based on this result, we can observe that if keyword number in the dictionary increases, SPEC achieves better performance than previous strategies. In other words, when the dictionary becomes larger, our proposed system achieves better performance than the two existing systems.

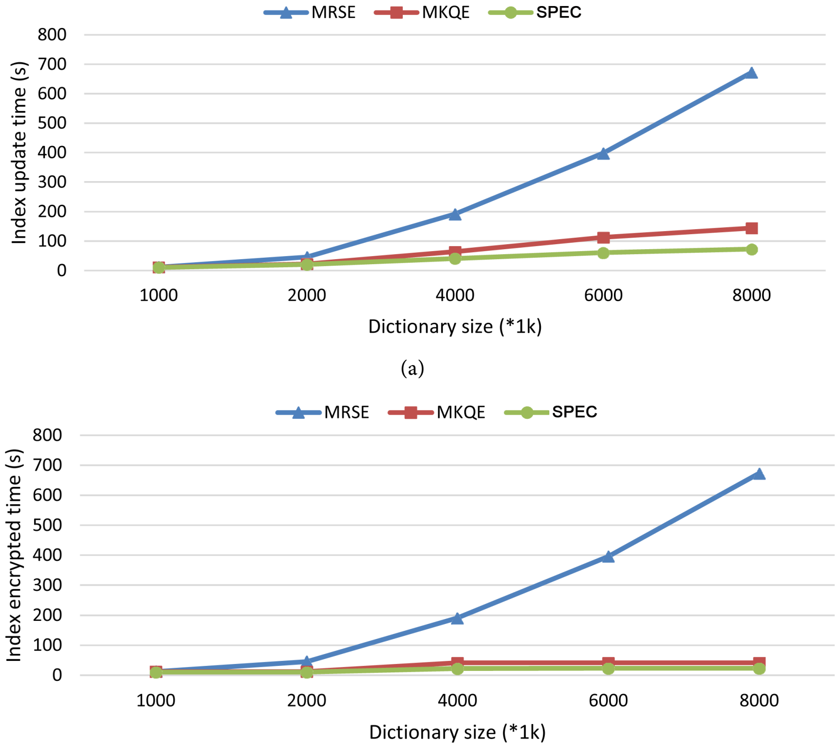
According to Figure 4(b) a comparison is held between the time consumption to generate the encrypted file indexes with different sizes of keyword dictionary. As shown in the results, SPEC takes less time to generate the indexes in all sce-



(a)

(b)

Figure 3. (a) The generation time of index for single file v.s. # of the keywords; (b) Time consumption comparison on trapdoor generation (ms).



(a)

(b)

Figure 4. (a) update index time starting with 1000 with update index; (b) Index encryption time comparison with 1000 file indexes encrypted.

narios. As per Figure 4(b), we can see that if the number of keywords in the dictionary becomes larger, the time of encryption index increases gradually and then SPEC outperforms all of MKQE and MSRE.

5.3. Keyword access Frequency Analysis

In this set of experiments, we compare the query results in SPEC, MRSE, and MKQE when taking the access frequency of keyword into account. The keywords that have high access frequency appear in the top k position in the matching result set. When we set k = 40, it means that the first 40 files having the highest scores will be returned for each query.

Figure 5 shows the results of the keywords that have high access frequency or Wight appear in the top k position in the matching result set and also the results search are ranked based on the history and the number of times the document id is present in the buckets as well as results query must be rank based on rank when return to the user as search results.

We can conclude from Figure 5. With take into account popular a keyword, our proposed system will achieve better performance than the existing systems, a keyword with a larger access frequency has a higher probability to appear in result set.

6. Conclusion

In the present paper, our main objective is to find an effective solution to the problems of multi-keyword ranked query over encrypted cloud computing. First of all, we gave a definition or a formulation of the problem, to analyse the solutions in hand and then we will use a new scheme called (SPEC) in order to solve this problem and therefore improve the performance of cloud computing and to safeguard privacy of data in comparison to the results of previous researches in regard to accuracy, privacy, security, key generation, storage capacity as well as trapdoor, index generation, index encryption, index update, and finally files retrieval depending on access frequency. Then, we have designed a new trapdoor generation algorithm, which may be able to solve finally out-of-order problem in

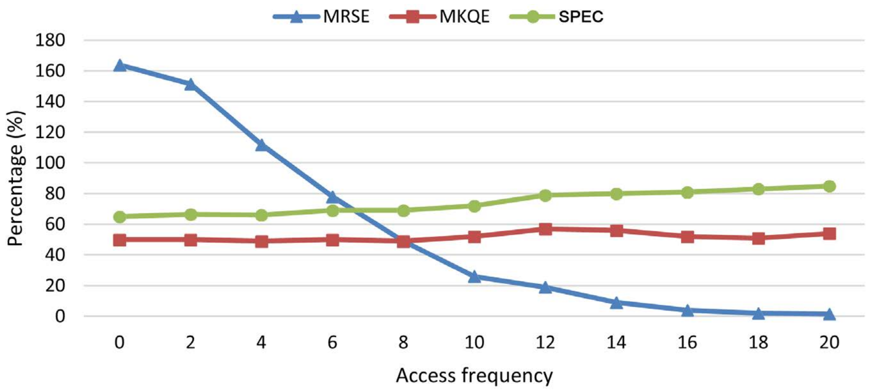


Figure 5. Percentage of files containing highest access frequency. Keywords in the top 20 locations.

the returned result set without affecting the accuracy and privacy of data. Moreover, the access frequency of keywords is considered as well in ranking algorithm when generating a query and must be rank based on rank. We have used Tripple Data Encryption Standard (TDES) and Advanced Encryption Standard (AES-128) Algorithms in order to encrypt files with the aim of ascertaining data privacy. We confirm that DC will be highly capable of retrieving files they really need. In view of simulation experiments, we can finally reveal that our model can be of better performance than previous ones and will have a good security level as well.

References

[1] Wang, C., Ren, K., Yu, S. and Urs, K.M.R. (2012) Achieving Usable and Privacy-Assured Similarity Search over Outsourced Cloud Data. *INFOCOM*, Orlando, 25-30 March 2012, 451-459. https://doi.org/10.1109/infcom.2012.6195784

[2] Buyya, R. and Dastjerdi, A.V. (2016) Internet of Things: Principles and Paradigms. Elsevier, New York.

[3] Rong, C., Nguyen, S.T. and Jaatun, M.G. (2013) Beyond Lightning: A Survey on Security Challenges in Cloud Computing. *Computers & Electrical Engineering*, **39**, 47-54.

[4] Zhang, R., Liu, J., Han, Z. and Liu, L. (2011) RBTBAC: Secure Access and Management of EHR Data. *International Conference on Information Society*, London, 27-29 June 2011, 494-499.

[5] Zhang, R., Liu, L. and Xue, R. (2014) Role-Based and Time-Bound Access and Management of EHR Data. *Security and Communication Networks*, **7**, 994-1015. https://doi.org/10.1002/sec.817

[6] Nabeel, M., Bertino, E., Kantarcioglu, M. and Thuraisingham, B. (2011) Towards Privacy Preserving Access Control in the Cloud. 7*th International Conference on Collaborative Computing*: *Networking*, *Applications and Work Sharing*, Orlando, 15-18 October 2011, 172-180. https://doi.org/10.4108/icst.collaboratecom.2011.247061

[7] Nabeel, M. and Bertino, E. (2012) Privacy Preserving Delegated Access Control in the Storage as a Service Model. 13*th International Conference on Information Reuse and Integration*, Las Vegas, 8-10 August 2012, 645-652.

[8] Nabeel, M. and Bertino, E. (2014) Privacy Preserving Delegated Access Control in Public Clouds. *IEEE Transactions on Knowledge and Data Engineering*, **26**, 2268-2280. https://doi.org/10.1109/TKDE.2013.68

[9] Nabeel, M., Shang, N. and Bertino, E. (2013) Privacy Preserving Policy-Based Content Sharing in Public Clouds. *IEEE Transactions on Knowledge and Data Engineering*, **25**, 2602-2614. https://doi.org/10.1109/TKDE.2012.180

[10] Raykova, M., Zhao, H. and Bellovin, S.M. (2012) Privacy Enhanced Access Control for Outsourced Data Sharing. *International Conference on Financial Cryptography and Data Security*, Kralendijk, 27 Februray-2 March 2012, 223-238. https://doi.org/10.1007/978-3-642-32946-3_17

[11] Kaur, G. and Mahajan, M. (2013) Analyzing Data Security for Cloud Computing Using Cryptographic Algorithms. *International Journal of Engineering Research and Applications*, **3**, 782-786.

[12] Zissis, D. and Lekkas, D. (2012) Addressing Cloud Computing Security Issues. *Future Generation Computer Systems*, **28**, 583-592.

[13] Han, J., Susilo, W. and Mu, Y. (2013) Identity-Based Data Storage in Cloud Computing. *Future Generation Computer Systems*, **29**, 673-681.

[14] Boutet, A., Frey, D., Guerraoui, R., Jégou, A. and Kermarrec, A.M. (2016) Privacy-Preserving Distributed Collaborative Filtering. *Computing*, **98**, 827-846. https://doi.org/10.1007/s00607-015-0451-z

[15] Fu, Z., Wu, X., Guan, C., Sun, X. and Ren, K. (2016) Toward Efficient Multi-Keyword Fuzzy Search over Encrypted Outsourced Data with Accuracy Improvement. *IEEE Transactions on Information Forensics and Security*, **11**, 2706-2716. https://doi.org/10.1109/TIFS.2016.2596138

[16] Fahl, S., Harbach, M., Muders, T. and Smith, M. (2012) Confidentiality as a Service—Usable Security for the Cloud. 11*th International Conference on Trust*, *Security and Privacy in Computing and Communications*, Liverpool, 25-27 June 2012, 153-162.

[17] Harbach, M., Fahl, S., Brenner, M., Muders, T. and Smith, M. (2012) Towards Privacy-Preserving Access Control with Hidden Policies, Hidden Credentials, and Hidden Decisions. 10*th Annual International Conference on Privacy*, *Security and Trust*, Paris, 16-18 July 2012, 17-24. https://doi.org/10.1109/pst.2012.6297915

[18] Kuzu, M., Kantarcioglu, M., Thuraisingham, B., Khan, L. and Schweitzer, H. (2013) Practical Privacy Preserving Record Integration and Search. The University of Texas, Austin.

[19] Wang, C., Cao, N., Ren, K. and Lou, W. (2012) Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data. *IEEE Transactions on Parallel and Distributed Systems*, **23**, 1467-1479. https://doi.org/10.1109/TPDS.2011.282

[20] Singh, M. and Singh, N.S. (2014) Implementation of Triple Data Encryption Standard Using Verilog. *International Journal of Advanced Research in Computer Science and Software Engineering*, **2277**, 667-670.

[21] Kakkar, A., Singh, M.L. and Bansal, P.K. (2012) Comparison of Various Encryption Algorithms and Techniques for Secured Data Communication. *International Journal of Engineering and Technology*, **2**, 87-92.

[22] Cao, N., Wang, C., Li, M., Ren, K. and Lou, W. (2014) Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data. *IEEE Transactions on Parallel and Distributed Systems*, **25**, 222-233. https://doi.org/10.1109/TPDS.2013.45

[23] Kamara, S. and Lauter, K. (2010) Cryptographic Cloud Storage. *International Conference on Financial Cryptography and Data Security*, Tenerife, 25-28 January 2010, 136-149. https://doi.org/10.1007/978-3-642-14992-4_13

[24] Fu, Z., Ren, K., Shu, J., Sun, X. and Huang, F. (2016) Enabling Personalized Search over Encrypted Outsourced Data with Efficiency Improvement. *IEEE Transactions on Parallel and Distributed Systems*, **27**, 2546-2559. https://doi.org/10.1109/TPDS.2015.2506573

[25] Han, F., Qin, J., Zhao, H. and Hu, J. (2014) A General Transformation from KP-ABE to Searchable Encryption. *Future Generation Computer Systems*, **30**, 107-115.

[26] Chen, R., Mu, Y., Yang, G., Guo, F., Huang, X., Wang, X. and Wang, Y. (2016) Server-Aided Public Key Encryption with Keyword Search. *IEEE Transactions on Information Forensics and Security*, **11**, 2833-2842. https://doi.org/10.1109/TIFS.2016.2599293

[27] Liu, Q., Wang, G. and Wu, J. (2012) Secure and Privacy Preserving Keyword Searching for Cloud Storage Services. *Journal of Network and Computer Applications*, **35**, 927-933.

[28] Shiba Sampat Kale, P. and Lahane, S.R. (2014) Privacy Preserving Multi-Keyword Ranked Search with Anonymous ID Assignment over Encrypted Cloud Data. *International Journal of Computer Science and Information Technologies*, **5**, 7093-7096.

[29] Xia, Z., Chen, L., Sun, X. and Wang, J. (2013) An Efficient and Privacy-Preserving Semantic Multi-Keyword Ranked Search over Encrypted Cloud Data. *Advanced Science and Technology Letters*, **31**, 284.

[30] Madane, S.A. and Patil, B.M. (2015) Comparison of Privacy Preserving Single-Keyword Search and Multi-Keyword Ranked Search Techniques over Encrypted Cloud Data. *International Journal of Computer Applications*, **126**, 34-38.

[31] Barde, C.R., Katkade, P., Shewale, D. and Khatale, R. (2014) Secured Multiple-Keyword Search over Encrypted Cloud Data. *International Journal of Emerging Technology and Advanced Engineering*, **4**, 528-532.

Abbreviations

D: The plaintext document collection expressed as a set of m documents D = $\{d_1, d_2\ldots d_m\}$.

C: The encrypted document collection, for D stored in the cloud server, expressed as c = $\{c_1, c_2\ldots c_m\}$.

R: relevant files after the match with a query that send from user R = $\{r_1, r_2\ldots r_k\}$.

W: the dictionary, including n keywords extracted from D, expressed as W = $\{w_1, w_2\ldots w_n\}$.

Ŵ: a subset of W, representing keywords extracted from D, expressed as Ŵ = $\{\hat{W}_{i1}, \hat{W}_{i2}\ldots \hat{W}_{il}\}$.

I: the searchable index in the entire document set D.

Tw$_i$: the trapdoor for search request of keywords Wi.