Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 25 (2016) 310 – 317

Procedia Technology

Global Colloquium in Recent Advancement and Effectual Researches in Engineering, Science and Technology (RAEREST 2016)

An efficient privacy preserving search scheme with access control for cloud data centers

Tresa Mary George V*, Shamna S, Jubilant J. Kizhakkethottam

Department of Computer Science and Engineering, Musaliar College of Engineering and Technology, Pathanamthitta 689653, India

Abstract

The internet and the emergence of social networks produce terabytes of data every day. In this big data scenario, the ability to outsource the data to a cloud storage facility saves the data management and storage facility cost. Some major challenges with this scheme are providing security and ensuring the privacy of the outsourced data. Although data security can be achieved through encryption, searching on encrypted data become a complex task. The proposed work suggests an efficient searching scheme for encrypted cloud data based on hierarchical clustering of documents. The hierarchical clustering method preserves the semantic relationship between the documents in the encrypted domain to speed up the search process. Consequently, the proposed system has linear computational complexity during the search phase in response to an exponential increase in the number of documents. The system also ensures data privacy by providing only limited access of the documents to the different types of users by implementing access control mechanisms resulting in more secured data storage in the cloud.

Peer-review under responsibility of the organizing committee of RAEREST 2016

Keywords: searchable encryption; multi keyword search; hierarchical clustering; access control

1. Introduction

A fundamental application of cloud computing is the ability to outsource remote data to external cloud servers to enable scalable data storage. The cloud server can provide a huge storage space and high computational power [1].

* Corresponding author.
E-mail address: vtresamg@gmail.com

Peer-review under responsibility of the organizing committee of RAEREST 2016
doi:10.1016/j.protcy.2016.08.112

Accordingly, enterprises and users who own a large amount of data can overcome their hardware limitations. As this technique is becoming more and more popular, the data volume in cloud storage facilities is experiencing a dramatic growth.

A major concern regarding the use of cloud computing for data storage is that, the outsourced data may contain sensitive information, such as photos, emails, bank statements etc. If the data is stored in a public cloud which is accessible to several other people without efficient protection mechanism, it can lead to severe privacy and confidentiality violations [2]. The traditional way to prevent sensitive data is encryption. The documents are encrypted before outsourcing them to the cloud. This however introduces further complexities during the search operation on encrypted data when legitimate users need access to those documents. Many researchers have investigated on this issue in the recent days and proposed several ciphertext search schemes based on cryptography techniques [3] [4]. However, these methods need extensive computations and suffer from high time complexity. Hence these methods are not suitable for a big data environment [5]. Another major drawback is that, the relationship between the documents is concealed during the encryption process. Maintaining such a relationship is important as it represents the properties of the documents.

It is also necessary to provide controlled access to the outsourced cloud data to different classes of users. The system must prevent unauthorized users from uploading corrupted documents to the cloud server. For example, consider a university cloud in which the student mark lists are stored in the cloud. In such a scenario, the students must be prevented from uploading their own mark lists thereby overwriting the original copy. To prevent this, the system will provide only download privileges to the student users of the cloud. Proper implementation of access control mechanisms will ensure such limited access to the different class of cloud users.

The proposed system uses a searching scheme based on multi keyword ranked search. In addition, a hierarchical clustering method is used to cluster the documents based on a relevance score. There is also a limit on the maximum size of each cluster. If the size of a cluster exceeds this limit, the cluster is further divided into sub clusters until the size of each cluster fall below the threshold value. During the search phase, the system iteratively determines the most relevant cluster. Only those documents in that cluster need to be searched, thereby it reduces the overall search time.

2. Related works

Many researches have proposed several methods for search on encrypted data in the cloud. Some of them and their drawbacks are discussed below.

2.1. Searchable encryption based on single keyword

In the method proposed by Song et al, [6] each word in the document is encrypted independently. This requires scanning of the entire data collection word by word. The major drawback of this method is the high search cost resulting from the scanning of entire document. Cash et al. [7] proposed a symmetric searchable encryption scheme. Though it provides high efficiency for large databases, it lacks a rank mechanism. If a large number of documents contain the searched keyword, the user has to manually select what they actually want, which in turn increase the overall search time.

2.2. Searchable encryption based on multiple keywords

Cao et al [8] proposed an architecture which perform multi-keyword search and also support result ranking by using k-nearest neighbor algorithm. However, the search time of this method grows exponentially in response to an exponentially increasing size of the document collections. Sun et al [9] proposed a new architecture. Though it provides better efficiency, the relevance between the documents is ignored and hence it does not return the most relevant results.

2.3. Boolean Symmetric Searchable Encryption

Tarik Moataz and Abdullatif Shifka [10] proposed a system for searching multiple keywords over encrypted data using Boolean Symmetric Searchable Encryption (BSSE). It uses Gram-Schmidt process to optimize the search process. It considers arbitrary boolean expressions such as conjunctions and disjunctions of keywords and their complement on keywords.

2.4. Fuzzy Keyword Search

The above mentioned searching schemes will retrieve files only based on exact match of the keyword. Any typos and inconsistencies in the format will not return the required documents. J. Li et al. [11] proposed a wild card based technique to create efficient fuzzy keyword sets that can be used for matching relevant documents. Whenever the exact match search fails, the search result is provided based on the fuzzy keyword data set.

3. System model and problem formulation

The proposed system uses a vector space model in which every document is represented by a vector. Every document can be seen as a point in a high dimensional space. The documents are classified into categories by using a clustering method. The proposed system uses a hierarchical clustering index, i.e. a hierarchy of clusters at different levels is used. Each cluster has a constraint on the minimum relevance score between the documents in that cluster. When a new document is added to the cluster, the constraint may get broken. In such a case, a new cluster center will be added to the system. After that, all the cluster centers will be re selected and all the documents will be reassigned. The maximum size of the cluster is also fixed for each level. If the size of a cluster exceeds the maximum limit, that cluster will be divided into multiple sub-clusters. When a search is being performed, only those documents in the relevant clusters need to be searched, thereby it reduces the overall search time.

During the search phase, the relevance score between the search query and the cluster centers of the first level index is computed. The cluster center with maximum relevance score will be selected and this process will be iteratively repeated for the children in the next level clusters until the smallest cluster in the lowest level is found. If this cluster does not contain the desired document, the system will trace back to the parent of the smallest cluster. This process is repeated until the desired document is found or the root cluster is reached.

3.1. System architecture

The system architecture is composed of mainly four entities as shown in Fig. 1. They are the data owner, the data user, the cloud server and the cloud manager. The data owner is the module responsible for collecting documents, performing the encapsulation, building the document index and outsourcing the encrypted document to the cloud server. The data user is the consumer of the documents and they must have necessary authorization before accessing this data. The cloud server is the entity which provides a huge storage space and necessary computational resources for the ciphertext search. The cloud manager is responsible for ensuring access control. It blocks all unauthorized requests for the data by checking the privacy settings of each user. When the cloud server receives a request for a document, this request is verified by the cloud manager. Upon successful verification, the cloud server returns the required documents.

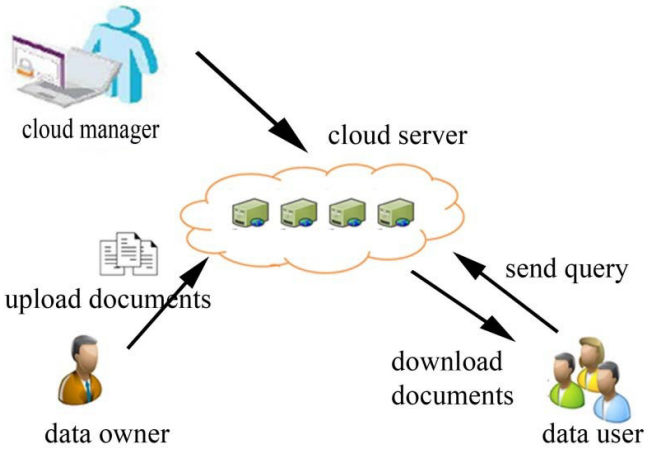

Fig. 1. System architecture

4. Implementation details

4.1. MRSE-HCI architecture

The proposed system uses Multi-keyword Ranked Search over Encrypted data based on Hierarchical Clustering Index (MRSE-HCI) scheme in which the vector space model is adopted from the Multi-keyword Ranked Search over Encrypted data (MRSE) [8] and the indexing is based on Hierarchical Indexing Structure (HCI) [12]. The detailed description is as follows. Every document is indexed by a vector and each dimension of the vector refers to a keyword. The value of each dimension indicates whether the keyword appears in the particular document. The query is also represented in a similar way as a vector. The lengths of the document vectors are normalized and hence the distance of points in the n-dimensional space reflects the relevance of corresponding documents. During the search phase, the cloud server component computes the relevance score between the query vector and the documents vector by computing their inner product. When the documents are stored in the cloud in an encrypted form, the semantic relationship between the documents will be lost. However the proposed system uses a clustering method. In the n-dimensional space, the points of highly relevant documents are very close to each other thereby, the semantic relationship between the documents is preserved.

When the volume of data in the cloud experiences a dramatic growth, the traditional search approaches will be very inefficient and has an exponential growth. To improve the search efficiency, a hierarchical clustering method is used. The hierarchical approach clusters the documents based on the relevance score at different levels. When the size of the cluster reaches the maximum cluster size threshold, the system partitions the clusters into sub-clusters until the criterion is satisfied. When the documents are being uploaded, the data owner also builds an encrypted index. A symmetric key encryption algorithm is used and the documents are encrypted using some random numbers and a secret key. When the data user needs a particular document, a query is submitted to the cloud server. The cloud server will return the target document to the data user.

The functions of the different components are described below.

Keygen: This function will generate the secret key sk used to encrypt the index and the documents. For this, a $(n + u + 1)$ bit vector S in which each element is an integer 1 or 0 and two invertible $(n + v + 1) \times (n + v + 1)$ matrices *M1* and *M2* whose elements are random integers are generated.

Index: This phase generates the encrypted index by using the above generated secret key. The clustering process

also takes place in this phase. The index algorithm is as follows.

1) A tokenizer and a parser tools are used to extract all the keywords present in the document.
2) The documents are transformed into a collection of Document Vectors (DV).
3) A Quality Hierarchical Clustering (QHC) method is used to generate the information about Documents Classification (DC) and the collection of Cluster Centers Vectors (CCV)$\{c_1 \cdots c_n\}$.
4) The data owner performs the dimension-expanding and vector splitting procedure on every document vector.
    a. During dimension-expanding procedure, each vector in CCV is extended to $(n + v + 1)$ bit-long vector, where the value in $n + j(0 \leq j \leq v)$ dimension is an integer number generated randomly and the last dimension is set to 1.
    b. During the vector splitting procedure, every extended document vector is split into two $(n + v + 1)$ bit-long vectors V' and V'' using the above generated $(n + v + 1)$ bit vector S as a splitting indicator.

Encryption: The plain document set D is encrypted using any secure symmetric encryption algorithm such as AES. The encrypted document is then outsourced to the cloud.

Trapdoor: When a user submits a query, the cloud manager will analyse the query and verify that the request come from an authenticated user. The keywords in the query are analyzed with the help of dictionary DW and a query vector QV is generated, which is then extended to a $(n + v + 1)$ bit vector.

Search: When the cloud server receives the query vector, the relevance score between the query vector and index vector of clusters are computed in a hierarchical manner. It finally choses the cluster with maximum relevance score as the target cluster and search for the required document. If the document is not found, it back tracks and choose a different cluster with next highest score. This process is repeated until the target document is found.

Decryption: This component is used by the data user to decrypt the returned document. The secret key is exchanged to the user through a secure mechanism.

4.2. Relevance measure

In the proposed system, the concept of coordinate matching is used as a relevance measure. The relevance score between document *di* and query q_w is determined as described in Equation 1.

$$R_{qdi} = \sum_{t=1}^{n+v+1} (q_{w,t} \times d_{i,t}) \tag{1}$$

The relevance score between query q_w and cluster center $lc_{i,j}$ is determined as described in Equation 2.

$$R_{qci} = \sum_{t=1}^{n+v+1} (q_{w,t} \times lc_{i,j,t}) \tag{2}$$

The relevance score between document $d_i$and d_j is determined as described in Equation 3.

$$R_{ddi} = \sum_{t=1}^{n+v+1} (d_{i,t} \times d_{j,t}) \tag{3}$$

4.3. Quality Hierarchical Clustering Algorithm

Some of the most widely used and popular clustering algorithms are *K-means* and *K-medoids*. In these algorithms, the value of *k* is fixed earlier. However, in a big data scenario, it is impossible to predict the value of k early. The clusters are to be generated dynamically. Hence a *dynamic K-means algorithm* is used. To keep the clusters dense and compact, a minimum relevance threshold value is maintained. While performing the clustering process, the relevance score between each document and its cluster center is computed and if this value is less than the minimum threshold value, a new cluster is added and all the documents are reassigned accordingly. This procedure is executed iteratively until a stable value of *k* is reached.

4.4. Search Algorithm

To search for a particular document, the cloud server first needs to find the cluster that most match the query. The cloud server uses the cluster index I_c and an iterative procedure as described below to find the top matched cluster.

1) The cloud server first computes the relevance score value between query T_w and encrypted vectors of the first level cluster centers in cluster index I_c as described in Equation 2. It then chooses the i^{th} cluster center $I_{c,1,i}$ with the highest score.
2) For each child cluster centers of the above selected cluster center, the cloud server computes the relevance score between T_w and every encrypted vectors of child cluster centers, and finally gets the cluster center $I_{c,2,i}$ with the top score.

The above procedure is iterated until the ultimate cluster center $I_{c,1,i}$ in last level l is achieved.

5. Results and analysis

5.1. Search Efficiency

The efficiency of the system was tested with a two level clustering model. The number of operation needed for the entire search process can be computed as described in Equation 4. To increase the search efficiency the system uses a static dictionary of keywords which does not effectively contribute to the search process. The terms like 'for', 'and' etc. in the search query will be removed and a modified query vector will be constructed. The subsequent comparisons are made only with the modified query vector. Let *x* denote the size of the static dictionary, *w* denote the number of query keywords, *u* denote the number of keywords in the modified query vector, *n* denote the total number of documents in the documents collection, *k* denote the number of categories in the first level cluster and *t* denote the average number of documents in the subsequent cluster.

$$Operations(Search\ process) = w * x + (w - u)k + (w - u - 1)t \tag{4}$$

The number of operations required by a system without any clustering technique is described in Equation 5.

$$Operations(existing\ system) = w * x + (w - u)n \tag{5}$$

During the search step, the existing system compares the query vector with the entire documents collection whereas the proposed system compares it only with the relevant cluster leading to significant reduction in search time.

5.2. Performance analysis

To test the performance of the proposed system, an experimental setup was built as follows. An application

simulating the activities of a university was created. The cloud storage platform for the system was provided by the Google public cloud. The data owners of the system are

1. The university which owns the mark lists and certificates of all the passed out and presently studying students.
2. The college which uploads the sessional marks and other student specific documents of all the students

The data set for the performance analysis was built from the above mentioned types of documents. The system was tested with a linear increase in the number of documents and the corresponding search times were estimated. It is evident from Fig 2 that the proposed system outperforms the existing system without clustering. The system was also tested with an exponential growth in the number of documents. Fig 3 shows that the proposed system with clustering has a linear growth in search time while the system without clustering has an exponential growth in search time.

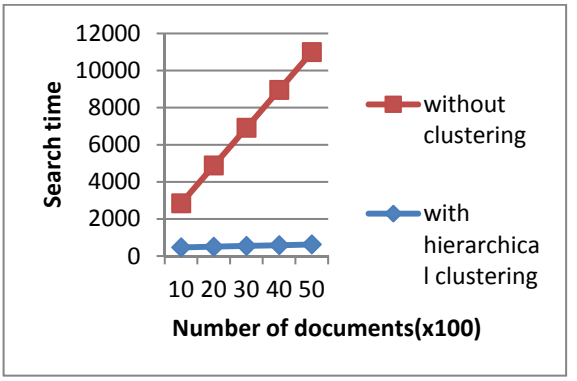

Fig. 2. Comparison of search time with a linear growth in documents collection

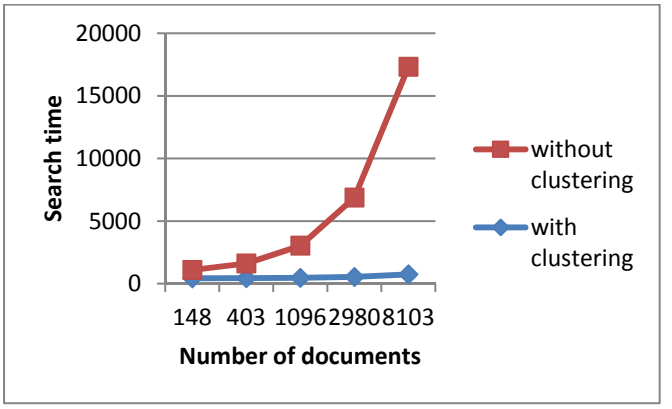

Fig. 3. Comparison of search time with an exponential growth in documents collection

5.3. Security analysis

A dedicated module called cloud manager is added to the proposed system to verify the authenticity of the arriving requests. To ensure the confidentiality and privacy of the documents stored in the cloud server, all the documents are encrypted using a symmetric encryption algorithm before uploading it to the cloud. In addition to that, the cloud storage provider also performs a two level encryption on the documents and returns a public key to the cloud manager. All the keys are managed by the cloud manager and only people with sufficient access rights can

decrypt the document. Consequently, the system ensures that even if an intruder accesses the document directly from the cloud server, they cannot get the plaintext of the documents.

6. Conclusion and future work

The problem of searching and securely accessing the encrypted data in the cloud is analyzed. It is understood that maintaining the semantic relationship between the documents reduce the search time for a document. The proposed work is based on multi keyword ranked search over encrypted data. The use of hierarchical clustering method to cluster the documents preserves the semantic relationship between the documents. The experimental results prove that the proposed system has a linear growth in time complexity when the size of the documents collection increases exponentially. It also implements a dedicated module named cloud manger to ensure the privacy of cloud data by granting only limited access to the documents collection to different classes of users. As future work, more secure algorithms can be developed for improving the privacy of the uploaded documents. More secure access control schemes such as Dynamic Information Flow Tracking (DIFT) techniques [13] with capabilities to recognize the advanced vulnerabilities can also boost up the overall performance of the system.

References

[1] Xian, C., Lu, Y. H., & Li, Z. (2007, December). Adaptive computation offloading for energy conservation on battery-powered systems. InParallel and Distributed Systems, 2007 International Conference on (Vol. 2, pp. 1-8). IEEE.

[2] Li, H., Dai, Y., Tian, L., & Yang, H. (2009). Identity-based authentication for cloud computing. In Cloud computing (pp. 157-166). Springer Berlin Heidelberg.

[3] Sun, W., Wang, B., Cao, N., Li, M., Lou, W., Hou, Y. T., & Li, H. (2013, May). Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking. In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security (pp. 71-82). ACM.

[4] Wang, B., Yu, S., Lou, W., & Hou, Y. T. (2014, April). Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud. InINFOCOM, 2014 Proceedings IEEE (pp. 2112-2120). IEEE.

[5] Sebastian, L. R., Babu, S., & Kizhakkethottam, J. J. (2015, February). Challenges with big data mining: A review. In Soft-Computing and Networks Security (ICSNS), 2015 International Conference on (pp. 1-4). IEEE.

[6] Song, D. X., Wagner, D., & Perrig, A. (2000). Practical techniques for searches on encrypted data. In Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on (pp. 44-55). IEEE.

[7] Cash, D., Jaeger, J., Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M. C., & Steiner, M. (2014, October). Dynamic searchable encryption in very-large databases: Data structures and implementation. In Network and Distributed System Security Symposium (NDSS'14).

[8] Cao, N., Wang, C., Li, M., Ren, K., & Lou, W. (2014). Privacy-preserving multi-keyword ranked search over encrypted cloud data. Parallel and Distributed Systems, IEEE Transactions on,25(1), 222-233.

[9] Sun, W., Wang, B., Cao, N., Li, M., Lou, W., Hou, Y. T., & Li, H. (2014). Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking. Parallel and Distributed Systems, IEEE Transactions on, 25(11), 3025-3035.

[10] Moataz, T., & Shikfa, A. (2013, May). Boolean symmetric searchable encryption. In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security (pp. 265-276). ACM.

[11] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. (2010) Fuzzy Keyword Search over Encrypted Data in Cloud Computing, Proc. of IEEE INFOCOM"10 Mini-Conference.

[12] Chen, C., Zhu, X., Shen, P., Hu, J., Guo, S., Tari, Z., & Zomaya, A. An Efficient Privacy-Preserving Ranked Keyword Search Method.

[13] Dalton, M., Kozyrakis, C., & Zeldovich, N. (2009, August). Nemesis: Preventing Authentication & Access Control Vulnerabilities in Web Applications. In USENIX Security Symposium (pp. 267-282).