INFORMATION TECHNOLOGY JOURNAL

Information Technology Journal 12 (24): 8176-8180, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.8176.8180
© 2013 Asian Network for Scientific Information

Hierarchical Data Retrieval Model For Big Data

Sijin Chen, Shao Bo Wu and Xue Ying Gao
School of Information and Communication Engineering,
Beijing Information Science and Technology University, Beijing, 100101, China

Abstract: Due to the massive data storage of big data, an efficient way of data retrieval which can improve the retrieval efficiency without compromising the security is necessary. Homomorphic Encryption allows the retrieval operated on encrypted data without decrypting them so that it can improve retrieval efficiency to a certain degree. But it still cannot satisfy the requirement for big data. In this study, we propose a distributed and hierarchical data retrieval model based on homomorphic encryption. This model can retrieve data according to the retrieval similarity level submitted by users which can improve the retrieval efficiency and in the last part of this study, we propose a way to enhance the retrieval security without compromising retrieval accuracy by adding obscure factor into the query information.

Key words: Cloud storage, hierarchical retrieval, retrieval similarity level, homomorphic encryption, obscure factor

INTRODUCTION

With the continuous growth of cloud computing, the era of big data has arrived. The amount of data stored in the cloud is soaring and data security and retrieval efficiency have become essential to big data. (Feng *et al*., 2011) CSA has released The Notorious Nine Cloud Computing Top Threats in 2013 (CSA, 2013) in which the top two is Data Breaches and Data Loss. So, how to guarantee the authorized users obtain the data they retrieve efficiently and accurately without causing security problems is of vital importance. The traditional data retrieval approach conducts the retrieval process on all of the data storage units which not only causes low efficiency but also lets the irrelevant servers obtain the query information which causes the risk of data breaches. (Song *et al*., 2000) Also, the retrieval result returned to users contains a lot of useless information which will cause the lack of accuracy. In the aspect of retrieval security, homomorphic encryption has been applied to cloud storage which allows the retrieval process operated on encrypted data directly and the retrieval result has been sorted before returned to users (Huang *et al*., 2010). This mechanism can enhance the retrieval security and improve the retrieval efficiency to a certain degree. However, more efficient retrieval approach is needed for the massive amount of information stored in big data. In this study, we propose a hierarchical (Qu *et al*., 2008) data retrieval model in which the query information will be compared with the retrieval information stored in index servers in different layers and the retrieval process will be conducted only on the data storage units which meet the retrieval requirements, the other data storage units will be eliminated. This retrieval model can improve the retrieval efficiency for big data. In the last part of this study, we also propose an algorithm to enhance retrieval security based on homomorphic encryption by adding obscure factor into the query information. This algorithm can make the attackers unable to obtin the correct query information even if they have managed to capture the encrypted query information. So, it can enhance the retrieval security to a certain degree.

HIERARCHICAL DATA RETRIEVAL MODEL

The hierarchical data retrieval model proposed in this study is demonstrated in Fig. 1.

In Fig. 1, user is the sponsor of a retrieval request. The query information will be processed by index servers in different layers. Then, after the query information has been processed by layers of index servers, some data storage units which meet the retrieval requirements will be locked and the retrieval process will be conducted only on these units. In this model, the number of layers of index servers is not fixed, it could be different according to the amount of data stored in the cloud. In this study, to demonstrate the basic principle, we set two layers of index servers which are index servers in upper layer and index servers in lower layer.

Corresponding Author: Sijin Chen, School of Information and Communication Engineering, Beijing Information Science and Technology University, Beijing, 100101, China

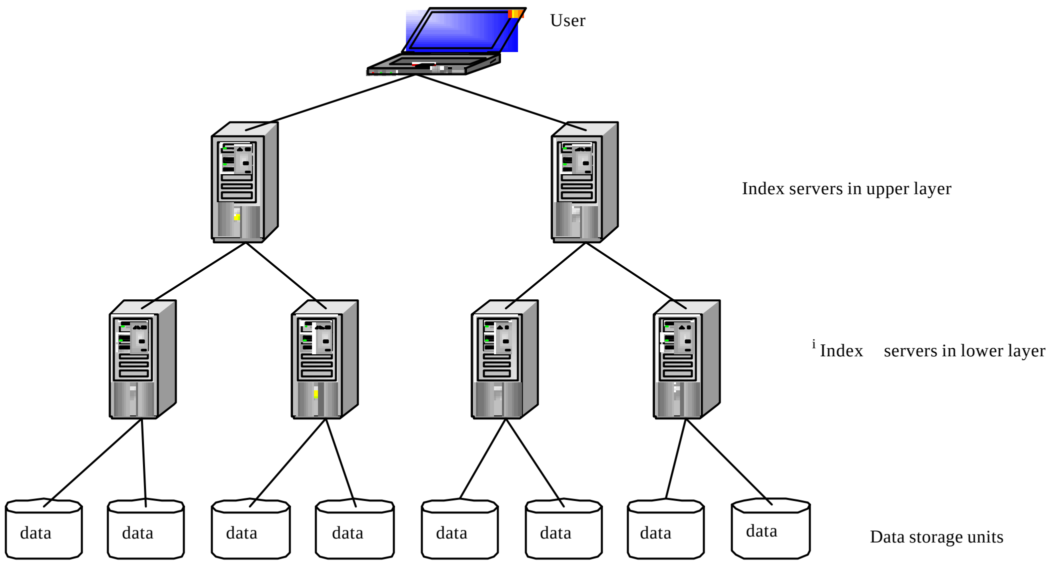

Fig. 1: Hierarchical data retrieval model

HIERARCHICAL DATA RETRIEVAL MODEL BASED ON SIMILARITY STORAGE

Similarity storage of data: About the data storage in Fig. 1, we propose a new mechanism to store the data: similarity storage of data. Data will be classified before they go into the data storage units. The procedure of data being stored into data storage units is described as below:

- K is a new created storage space which has n data storage units. A is a data set to be stored into K, K = (m_1, m_2... m_n) and A = (a_1, a_2... a_j). Before storing A into K, the retrieval information of A, i_{index}, will be extracted. i_{index} = (i_{index1}, i_{index2}... i_{indexj}). The index servers (which are index server NO.3,4,5,6 in Fig. 2) in lower layer will store these retrieval information and the data storage units will store the data set A
- B is a new data set to be stored into K. B = (b_1, b_2... b_j). Before storing B into K, the retrieval information of B will be extracted as well. B_{index1} = (b_{index1}, b_{index2}... b_{indexj}). Now index servers conduct AND operation between i_{index} stored in each index servers and B_{index}: i_{index} and B_{index}, then the data set B will be stored separately into the data storage units connected to the index servers which have the biggest similarity with B_{index}. Also the i_{index} stored in this index server will be updated to the union set of i_{index} and B_{index}. Here is an example of the procedure demonstrated by pseudo-code:

```
for (int q = 1,q<=j,q++)
if ($i_{index1}$ and $b_{index2}$>= $i_{indexq}$ and $b_{index2}$)
    //$b_{index2}$ and $i_{index1}$ have the biggest similarity.
$i_{index1} = i_{index1} \cup b_{index2}$
    // update $i_{index1}$
```

Then, data b_2 will be stored into the data storage units linked to the index server in which i_{index1} is stored and the index servers in upper layer will store the retrieval information stored in the index servers in lower layer. In Fig. 1, index server NO.1 will store the retrieval information stored in index server No. 3 and 4.

Whenever there are new data set need to be stored into K, the above steps will be repeated.

Hierarchical data retrieval mechanism based on retrieval similarity level: In the aspects of data retrieval, we propose a hierarchical data retrieval mechanism based on retrieval similarity level. This mechanism consists of three key modules: the index servers in upper layer, the index servers in lower layer and data storage units which are either real hardware storage units or storage units virtualized from the hardware by virtualization technology. (Liu, 2012)

The mechanism proposed in this study is based on retrieval similarity level and the realization of retrieval similarity level is demonstrated as below:

- The index servers in upper layer will conduct AND operation between the query information Q and the

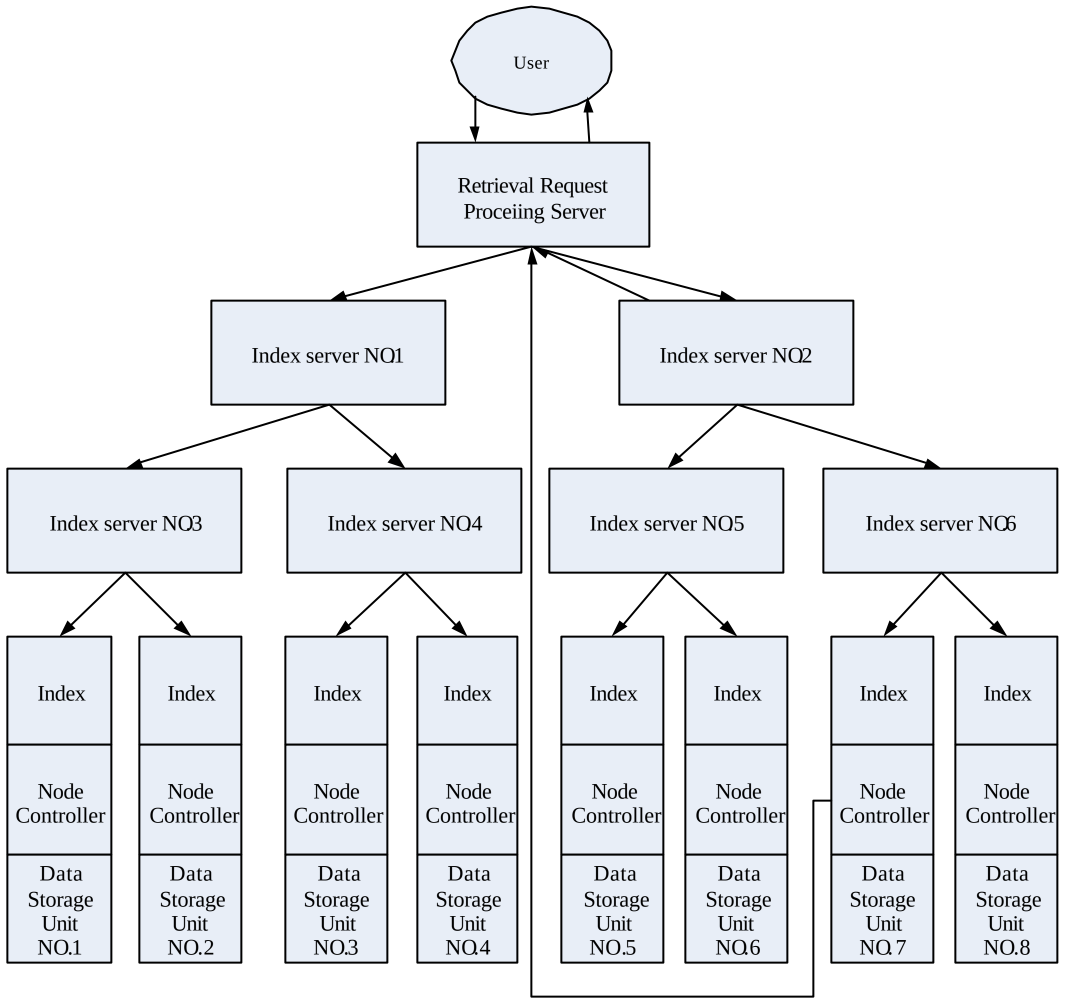

Fig. 2: Hierarchical data retrieval model

retrieval information stored in the index servers. The result of the AND operation is a, the similarity factor. A = (a_1, a_2... a_p), in which p is the number of index servers in upper layer. (in Fig. 2, p = 2 and the AND operation will be conducted between Q and the retrieval information stored in index servers No.1 and 2 and the similarity factor is a_1 and a_2)

- The user can submit a similarity requirement. The requirement could be a half-open interval which will be returned to the retrieval request processing server
- After obtaining the similarity requirement form the user, the retrieval request processing server will determine on which data storage units the retrieval information will be conducted. In Fig. 2, assuming that a_1 = 50%, a_2 = 80% and the similarity requirement submitted by the user is >70%, then the retrieval request processing server will eliminate the index server No. 1 and the retrieval operation will be conducted on the data storage units which are controlled by index server No. 2

Figure 2 is an example that illustrates the procedure of hierarchical data retrieval mechanism.

The specific retrieval process is demonstrated as below. The number is consistent with the number in Fig. 2:

- On receiving the retrieval request form the user, the retrieval request processing server operates partition processing in word field (Song *et al*., 2000) to the searching term to obtain the query information. Then the query information will be encrypted and becomes the encrypted query information Q

- The retrieval request processing server will send Q to the index servers in upper layer, index server No. 1 and index server No. 2
- Index server No. 1 and 2 will compare Q with the retrieval information stored in them. Because Homomorphic Encryption is applied here, the comparison operation can be conducted without decrypting Q or the retrieval information stored in index servers. Assuming that, the result of comparison between Q and the retrieval information stored in index server No. 1 does not meet the requirement of retrieval similarity level and the result of comparison between Q and index server No. 2 does. Then, the retrieval process will be conducted only on data storage units No. 5 to 8 or part of them. The data storage units No. 1 to 4 will be eliminated
- Index server No. 2 sends Q to the index servers to which it is connected in lower layer, index server No. 5 and 6
- Index server No. 5 and 6 will compare Q with the retrieval information stored in them. Assuming that the result of comparison between Q and the retrieval information stored in index server No. 5 does not meet the requirement of retrieval similarity level and the result of comparison between Q and index server No. 6 does. Then the retrieval process will be conducted only on data storage units No. 7 and 8 or one of them. The data storage units No. 5 and 6 will be eliminated
- Index server No. 6 will send Q to the data storage units which are controlled by it, data storage unit No. 7 and 8
- On receiving Q, data storage unit NO.7 and 8 will compare Q with the retrieval information stored in them. Then they will know whether the data stored in them meet the requirement of retrieval similarity level. Assuming that data storage unit NO.7 has stored the data which meet the retrieval requirement
- The retrieval process will be conducted on data storage unit No. 7
- The result of retrieval will be returned to the retrieval request processing server without being decrypted and then will be returned to user after sorted by the retrieval request processing server

Now, the retrieval operation in the cloud has been finished and the cloud has returned the result which meets the requirement submitted by user. User will obtain the plaintext result after decrypting the encrypted result using his own secret key.

ENCRYPTION ALGORITHM IN THIS MODEL

Homomorphic Encryption (HE): In traditional data retrieval mechanism, the encrypted date need to be decrypted before the retrieval operation is conducted and the retrieval result will be returned to user after being encrypted. This kind of mechanism has shortage in two aspects: low efficiency and the management of secret keys will cause security problems. (Li and Zhao, 2013) In this study, Homomorphic Encryption(HE) is applied which allows the retrieval process operated on encrypted data directly without decrypting them.
Homomorphic Encryption (HE) had been proposed by (Rivest *et al*., 1978). The basic principle is that, the result from some specific algebraic manipulation on the encrypted original data can be decrypted to the same plaintext as the algebraic manipulation operating directly on the unencrypted original data. (Qian and Wu, 2011) When Homomorphic Encryption (HE) is applied in cloud storage, it can realize that, the servers do not have to decrypt the encrypted query information and the encrypted data stored in data storage units, the retrieval process can be operated directly on the encrypted data. This mechanism can protect the sensitive information from data breaches. So, it can enhance the retrieval security to the third party.

Obscure factor: In order to enhance the retrieval security further, in this study we propose an algorithm to encrypt the retrieval request. Obscure factor is added into the retrieval request which can make the attacker unable to obtain the correct retrieval request even if he has managed to capture the encrypted query information (Orencik and Savas, 2012). In this study, the obscure factor is added into the retrieval request through a very simple way so that it does not compromise the retrieval efficiency very much.

We cut the query information Q into two parts which have the same length, b and c, then some bits of 0 will be added to the latter part of Q and the former part of c. Now we will get two pieces of new query information: d and e. Then, the retrieval process will be conducted based on d and e and the sum of the retrieval results will be returned to user. The final result improves to be the same as from the retrieval process without the obscure factor. Here is an example about how the obscure factor is added into the query information and how the result remains the same.

The first part of the example is based on JAVA:

- String q = “111001” ;
- String b = q.substring(0,2); //b = “111”

- String c = q.substring(3) ; //c = “001”
- String i = “000” ; //the obscure factor
- String d = b + i ; //d = 111000
- String e = i + c ; //e = 000001

Now, the obscure factor has been added into the query information.

Assuming that, i_{index} = 000111, so the result of the AND operation between Q and i_{index} is 1_0. After the obscure factor is added into Q, the result of the AND operation between d and i_{index} is 0 and that between e and i_{index} is 1, so the sum of the two results is 1. It turns out the obscure factor does not affect the retrieval accuracy at all. Even if the attackers have captured d and e and managed to decrypt them into 111000 and 000001, they still cannot obtain the correct retrieval request. Therefore, this algorithm is able to enhance the retrieval security without compromising the retrieval accuracy.

CONCLUSION

In this study, we propose a hierarchical data retrieval model based on homomorphic encryption. The retrieval process can be operated directly on the encrypted data which can lower the risks of data breaches. (China Telecom network security laboratory, 2012). The hierarchical model will eliminate not only the operation of retrieval which does not meet the requirements but also the work to sort the useless retrieval results, so it can improve the retrieval efficiency compared with the traditional retrieval mechanism. In the last part of this study, we propose an algorithm to enhance retrieval security. Obscure factor is added into the query information which can make the attackers unable to obtain the correct query information even if they have captured the query information with obscure factor in it. So, this model can improve data retrieval efficiency and enhance retrieval security for big data.

ACKNOWLEDGMENTS

This study was supported by Key Program of Beijing Municipal Natural Science Foundation (4131003) and National Natural Science Foundation of China (61271198) and National Natural Science Foundation of China (61370065) and Beijing Education Commission Science and Technology Development Plan (KM201311232010).

REFERENCES

CSA, 2013. The notorious nine: Cloud computing top threats in 2013. Cloud Security Alliance, Top Threats Working Group, pp: 6-7.

China Telecom Network Security Laboratory, 2012. Cloud Computing Security Technology and Application. Publishing House of Electronics Industry, China.

Feng, D.G., M. Zhang, Y. Zhang and Z. Xu, 2011. Study on cloud computing security. J. Software, 22: 71-83.

Huang, Y.F., J.L. Zhang and X. Li, 2010. Encrypted storage and its retrieval in cloud storage applications. ZTE Commun., 4: 33-35.

Li, W.C. and F.Y. Zhao, 2013. Enterprise data encryption and ciphertext fulltext retrieval on cloud storage. J. Chinese Comput. Syst., 2: 429-433.

Liu, P., 2012. Cloud Computing. 2nd Edn., Publishing House of Electronics Industry, China.

Orencik, C. and E. Savaa, 2012. An efficient privacy-preserving multi-keyword search over encrypted cloud data with ranking. Distrib. Parallel Databases. 10.1007/s10619-013-7123-9

Qian, P. and M. Wu, 2011. Survey of privacy preserving data mining methods based on homomorphic encryption. Appl. Res. Comput., 5: 1615-1616.

Qu, W., K. Li and M. Kitsuregawa, 2008. Enhance retrieval efficiency on the internet. Proceedings of the IEEE 7th International Conference on Networking, April 13-18, 2008, Cancun, Mexico, pp: 279-284.

Rivest, R.L., L. Adleman and M.L. Dertouzos, 1978. On data banks and privacy homomorphisms. Foundations of Secure Computation, 1978.

Song, D., D. Wagner and A. Perrig, 2000. Practical techniques for searches on encrypted data. Proceeding of the IEEE Symposium on Security and Privacy, May 14-17, 2000, Berkeley, CA., USA., pp: 44-55.