SECURE CLOUD DATA SHARING BY AGGREGATION OF KEYS INTO A SINGLE DECRYPTION KEY

Alanta Abraham[1], Shreenath Acharya[2]

[1]*Assistant Professor, Department of Computer Science, Don Bosco College, Kozhikode, Kerala, India*
[2]*Assistant Professor, Department of Information Science and Engineering, St Joseph Engineering College, Mangaluru, Karnataka, India*

Abstract

Cloud computing offers a number of advantages to all type of users including end users and business enterprises. It is a technology which allows the users to get access of applications and resources from anywhere which are not residing in our computer, but residing in some other locations. The problems in cloud computing are regarding the security of stored data as well as the shared data. To protect one's data over cloud, the basic techniques used include encryption, authentication and authorization which are becoming less efficient these days. Security and privacy issues get even worse when a set of particular data needs to be shared with specific users. Traditionally there are two main methods to share a set of selected data with another particular user. Either encrypt all data with single key and give the corresponding single decryption key to the other party, or encrypt each data with distinct keys and send corresponding keys to the receiver. In the first method, all the data will be leaked to the receiver. Obviously, the second method is inefficient since transferring and storing all the keys require large space. In this project, a set of data, called the ciphertext class is being shared with a particular user or group of users, as per the data owner's requirement. A single, compact, but powerful key is used as decryption key which is generated using a combination of different cryptographic algorithms as well as the ciphertext class identifier. The user who is receiving the ciphertext class will be able to decrypt it only if he is the indented user and he possesses the powerful decryption key. Hadoop framework is used for providing the distributed cloud environment. The implemented project works well for files of different types and sizes which are shared to individual users as well as groups of users. The results show that there is not much time difference for different types of files for encryption and uploading as well as for decryption and downloading.

Keywords: *Ciphertext Class, Encrypt, Hadoop*

----------***----------

1. INTRODUCTION

Cloud computing is everywhere. It assures to reduce operational and capital costs and let IT organizations to focus on strategic projects instead of keeping the datacenter running. In short, cloud computing is a technology which allows us to get access of applications and resources which are not residing in our computer, but residing in some other locations. Cloud computing provides a number of advantages both to end users as well as businesses of all sizes. Cost efficiency is the most important advantage of cloud computing, which can be achieved by avoiding investment in stand-alone software or services. Another advantage is the convenience and continuous availability. Public clouds provide services that are available wherever the user is located. The backup and recovery of data is simplified in cloud because those now reside not on a physical device, but on the cloud. Scalability is an important feature in cloud. Cloud services are deployed only when it is needed on a pay-as-you-go basis. Increased storage capacity is another advantage of cloud computing.

Cloud systems can be used for data sharing which will give great benefits for the user. The main benefit for organizations is higher productivity. Multiple users from different organizations can contribute to the data in cloud thereby reducing the time and cost. With social networking services such as Facebook, the benefits of sharing data are numerous. Google Docs provides data sharing capabilities as groups or teams. Cloud data sharing is beneficial in so many fields such as healthcare. However cloud is vulnerable to different kinds of privacy and security attacks.

Due to the emergence of social network sites such as facebook and e-commerce providing organizations, the amount of data generated daily is very large. One organization can collaborate the data provided by other organizations. So providing secure access to the shared data became a big issue. If the Cloud Service Providers (CSP) are untrusted, the risk increases. CSPs can behave unfaithfully for increasing their profit margin or something like that. CSPs may discard the rarely accessed data not in a timely fashion. Or, they may try to hide the incidents of data loss to keep up their reputation. Several cryptographic techniques have been introduced for solving this issue. But many of them were not that much useful in the case of efficiency and cost optimization. Besides, when a user wants to share a set of his private data with another person, the existing methods gives a number of issues. For example, he/she can encrypt all the data in the set using the same key and share the only one corresponding decryption key with the other party. This method will lead to the issue that data which are not meant to see by the receiving party also can be decrypted. In an alternate method, the user can encrypt each data in the set

with different keys and share each of the corresponding decryption keys with the receiving person. The problems with this method are the sharing of a number of decryption keys, their transmission, storage, etc. The proposed system makes use of a more efficient method to overcome these problems.

In this project, decryption key has been made more powerful so that it allows decryption of multiple ciphertexts without increasing its size, using an improved public key encryption. In this method, the ciphertexts are further categorized into different classes, called the ciphertext class which is the identifier of the ciphertext. Thus, users encrypt a message not only under a public-key, but also under the ciphertext class. The key owner holds a master-secret called master-secret key, by using which data owner can extract secret keys for different classes that can be an aggregate key which is as compact as a secret key for a single class, but sum totals the power of many such keys. This system addresses the above mentioned two problems of viewing unwanted data and transmission and storage of decryption keys. Also, only the particular specified person will be able to decrypt the files in the ciphertext class which is shared with him.

The advantages of proposed system are the delegation of decryption can be efficiently implemented with the aggregate key, which is of fixed size. It also results in large number of ciphertext classes (as one can reserve a large number of ciphertext classes because of rapid growth of amount of cloud data) and makes key management easy(since here it uses aggregation of a number of keys).

2. LITERATURE SURVEY

There has been a lot of works carried out on secure storage and sharing of cloud data. A progressive elliptic curve encryption (PECE) scheme is suggested [1][5][10]. In that, a piece of data is encrypted a number of times using a number of multiple keys and decrypted using only one key. Attribute Based Encryption (ABE), in which access control policy is defined and user/data attributes should satisfy it to get access, Proxy Re-encryption (PRE), in which data owner's private key is divided into two parts and hybrid ABE and PRE were also the well known approaches in this regard[1]. Most of these methods are based upon the cryptographic approaches mentioned in [2] such as AES, 3-DES etc.

A secure multi-owner data sharing scheme for dynamic groups in the cloud was suggested in [3] which enables every cloud users to share data by the untrusted cloud via group signature and broadcast encryption techniques. Identity based broadcast encryption [4][11] is an efficient method to share data securely to a group of users.

The authors of paper[5] proposed a mechanism to encrypt the data before storing into the cloud. When that data is to be shared, that encrypted data will be re-encrypted without being decrypted first. The re-encrypted data will be then cryptographically accessible only by the authorized users. There is also an important role for key management in secure data sharing as specified in [6].

A perfect decentralized access control scheme with aggregate key encryption for storing data in cloud has been proposed[7]. a number of contributions are done such as distributed access control of cloud data such that only authorized users can access them, authentication of users, user identity protection, decentralized architecture ie, there can be more than one KDCs for key management, aggregate key encryption which is highly secure, and so on.

The problem of handling keys in an access hierarchy, which is mostly in the form of a tree is discussed in paper [8]. The access hierarchy is represented as a set of classes of data. The owner can classify the data based on their subject, which can be modeled as a directed graph, in which each intermediate node indicates a class. A user who gets access to a particular class can get access to all of its descendent classes through deriving keys from the parent class.Hadoop and HDFS can be used to provide a common and distributed storage space[12].

3. PROBLEM STATEMENT

In cloud storage, data owners can let a number of selected users view a subset of their private data. For example, in Facebook we can customize who can view and share our private pictures. But the problem is how to effectively share the data. The existing cryptographic key assignment schemes have some disadvantages such as the cost of storing and transmitting ciphertext is more, increased cost in storage of secret keys and if the number of decryption keys increased, the cost and complexities will also increase. A Key Aggregate Cryptosystem (KAC) was developed by the authors of paper [13] which has solved this problem to an extent. But the limitation of that was the predefined bound of the number of ciphertext classes. Since the amount of data or number of ciphertexts increase rapidly in cloud storage the aim is to reserve enough ciphertext classes.

Let U be the set of users where U = u1,u2,...uk and N = n1,n2,...nk where each ni represents the number of ciphertext classes of user ui. Also let M be the set of plaintext messages, S be the set of ciphertext classes of a user and C be the set of ciphertexts generated after encryption. By using MK, a master-secret key and S, the aggregate key K for S is produced. Suppose M = m1,m2,.....mz; then ci can be generated using PK, i and mi. When someone wants to share a set S of ciphertexts with a user ui he/she can compute K by using MK and S and share it.

4. ARCHITECTURE

A decryption key has been made more powerful so that it allows decryption of multiple ciphertexts without increasing its size, using a new public key encryption called Key Aggregate Cryptography (KAC). For using KAC, the ciphertexts are further categorized into different classes which is the identifier of the ciphertext. Thus, users encrypt a message not only under a public-key, but also under the ciphertext class. The key owner holds a master-secret called a master-secret key, using which secret keys can be

extracted for each different classes that will be an aggregate key for decryption which is as compact as a secret key for a single class, but consists the power of many such keys. The advantages of proposed system are the delegation of decryption can be efficiently implemented with the aggregate key, which is of fixed size. It also results in large number of ciphertext classes (as one can reserve a large number of ciphertext classes because of rapid growth of amount of cloud data) and makes key management easy(since here uses aggregation of a number of keys).

There are four main modules in the proposed project as shown in Figure 1:

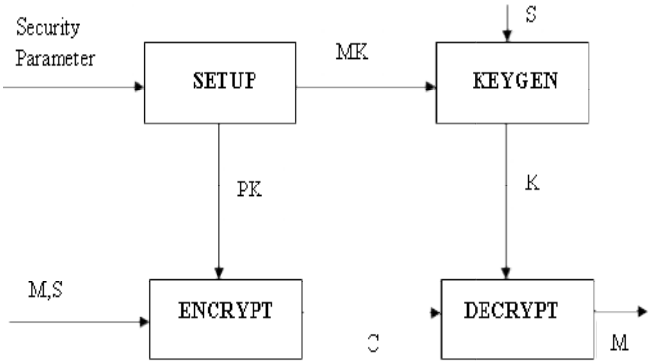

Fig 1: Modules of the System

where MK is theMaster Key, PK is the Public Key, K is the Aggregate Key, M and S are Messgage and Ciphertext class respectively and C is the Ciphertext.

- **Setup**: The setup algorithm takes no input other than the security level parameter. It outputs the public parameters PK and a master key MK
- **Encrypt**: Encrypt(PK,M,S). The encryption algorithm takes as input the public parameters PK, message M, and ciphertext class S. The algorithm will encrypt M and produce a ciphertext C such that only a user that possesses a set of ciphertext class S will be able to decrypt the message.
- **KeyGen** : Key Generation(MK,S). This key generation algorithm takes the master key MK and a set of attributes S that describe the key as inputs. It outputs a private key K
- **Decrypt**: Decrypt(PK, C, K). The decryption algorithm takes as input the public parameters PK, a ciphertext C, which contains an access policy S, and a privatekey K, which is a private key for a set S of attributes and returns the message M.

When a user wants to share a set of his data with another particular user, he creates a ciphertext class containing those selected information in encrypted form. During this, the aggregate decryption key which is used for decrypting all of the files in this particular ciphertext class also will be generated. When the other user receives the ciphertext class, he can decrypt it using the aggregate decryption key which is created using the public key, master key and the ciphertext identifier.

5. EXPERIMENTS AND EVALUATION

When a user login to the system an AES key and a RSA public/private key pair are generated for him for that login session. When he wants to share a selected set of files over the cloud with a particular user, each file are encrypted using the AES key before uploading to the cloud and these files are added to a ciphertext class. While sharing this ciphertext class with the other user, the AES key corresponding to this class is encrypted using his RSA public key. Then these keys together with the file ids of ciphertext class are mailed to the recipient. He can then decrypt and download the files by providing these credentials. A comparative study is carried out on the time taken for uploading and downloading of files based on type and size of files, which are depicted in the following tables.

Table 1: Time taken for encryption and uploading of different types of files of various sizes

file size (in kb)	doc (in sec approx)	docx (in sec approx)	odt (in sec approx)	txt (in sec approx)	pdf (in sec approx)
1-30	2.18	2.32	2.29	2.34	2.23
30-60	2.27	2.22	2.25	2.35	2.25
60-90	2.27	2.32	2.30	2.23	2.25
90-120	2.23	2.29	2.26	2.26	2.32
120-150	2.30	2.22	2.26	2.29	2.32

Table 2: Time taken for encryption and uploading of multimedia files of various sizes

file size (in mb)	mp3 (in sec approx.)	jpg (in sec approx.)
1	2.25	2.24
3	2.30	2.34
5	2.30	2.56

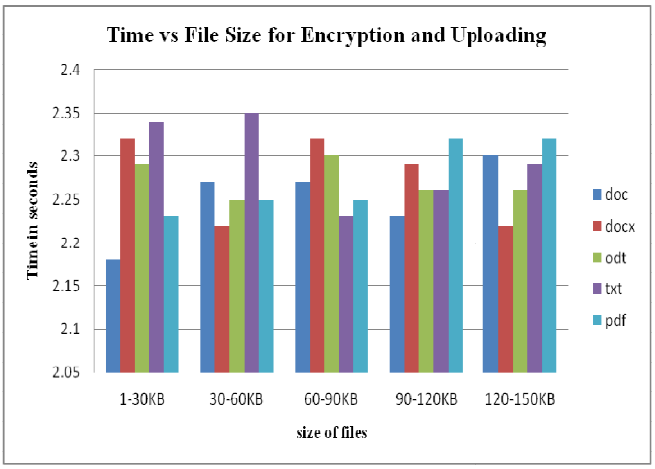

Fig2 : Graph Plotted for Time vs File Size for Encryption and Uploading

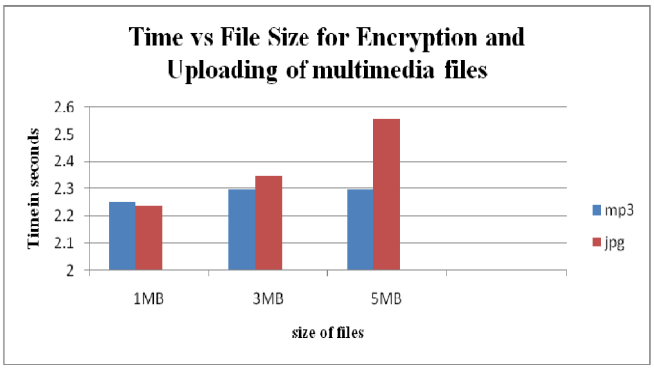

Fig 3 : Graph Plotted for Time vs File Size for Encryption and Uploading of Multimedia Files

From Table 1 and Fig 2 it is shown that for encrypting and uploading different types of files, the time taken is almost same for files which comes under same range of size. As the size increases time also varies accordingly. From Table 2 and Fig 3 which depict the time taken for encryption and uploading of multimedia files such as mp3 and jpg, it is understood that jpg files take more time than mp3 files for the same range of size and it varies accordingly as the size of files increase.

Table 3: Time taken for decryption and downloading of different types of files of various sizes

file size (in kb)	doc (in sec approx)	docx (in sec approx)	odt (in sec approx)	txt (in sec approx)	pdf (in sec approx)
1-30	1.20	1.16	1.26	1.06	1.25
30-60	1.13	1.26	1.11	1.17	1.12
60-90	1.15	1.15	1.10	1.13	1.27
90-120	1.09	1.13	1.15	1.20	1.13
120-150	1.16	1.13	1.17	1.15	1.12

Table 4: Time taken for decryption and downloading of multimedia files of various sizes

file size (in mb)	mp3 (in sec approx.)	jpg (in sec approx.)
1	1.28	1.21
3	1.20	1.09
5	1.17	1.17

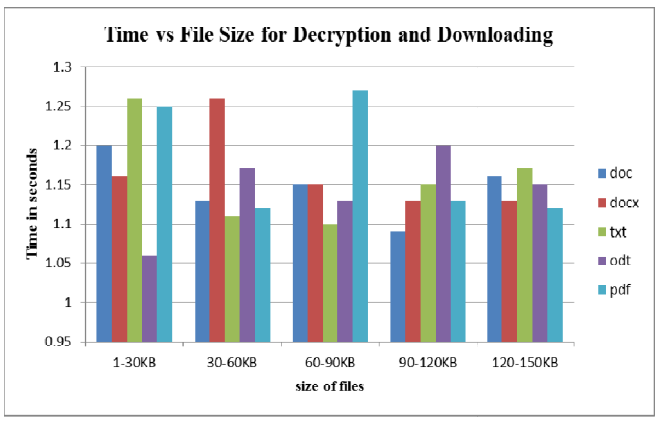

Fig 4: Graph plotted for Time vs File Size for Decryption and Downloading

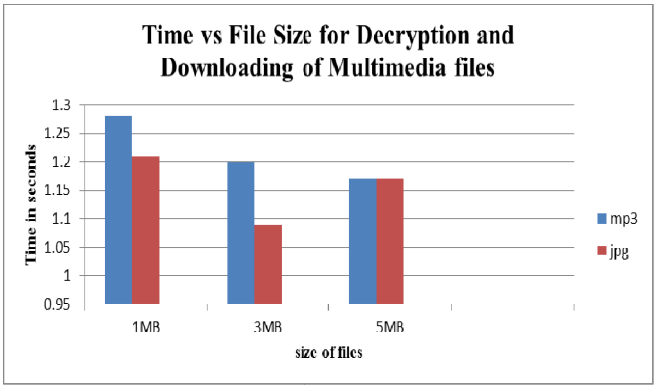

Fig 5: Graph plotted for Time vs File Size for Decryption and Downloading of Multimedia Files

From Table 3 and Fig 4, it is understood that for decrypting and downloading different types of files, the time taken is almost same for files which comes under same range of size, as in the case of uploading. As the size increases time also varies accordingly. From Table 4 and Fig 5, which depict the time taken for decryption and downloading of multimedia files such as mp3 and jpg, it is understood that jpg files take a little more time than mp3 files for the same range of size and it varies accordingly as the size of files increase and it takes less time to decryption than encryption.

6. CONCLUSION

In this project, a secure method for sharing a set of selected data with a particular user or group of users is modelled. It eliminates the drawbacks of the traditional ways used for sharing such data which includes the data privacy issues and problem of transferring and storing large number of decryption keys. Here the set of data is denoted as ciphertext class and for allowing a receiver for decryption, the single compact, powerful decryption key is used for every files in a ciphertext class. The key is composed of aggregation of some credentials such as class identifier and combination of keys of different cryptographic algorithms. It is found that, this mechanism is suitable for different types of files such as doc, docx, txt, jpg, mp3 etc. It is also clear from the obtained results that for all files which comes under same range of size, the time taken for uploading as well as downloading is almost equal. The slight changes in the observed time are based on the variation of file sizes.

REFERENCES

[1]. Danan Thilakanathan, Shiping Chen, Surya Nepal and Rafael A. Calvo "Secure Data Sharing in the Cloud", Springer-Verlag Berlin Heidelberg, 2014J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73.

[2]. Rashmi Nigoti, Manoj Jhuria and Dr.Shailendra Singh "A Survey of Cryptographic Algorithms for Cloud Computing", International Journal of Emerging Technologies in Computational and Applied Sciences,4(2), March-May 2013, pp.141-146.

[3]. S.Surya and V.Karuppuchamy "Secure Sharing Of Data for Dynamic Multi-Owner in Cloud Storage",International Journal of Innovative Research in Computer and Communication Engineering Vol.2, Special Issue 1, March 2014

[4]. Sherman S.M. Chow, Cheng-Kang Chu, Xinyi Huang Jianying Zhou, and Robert H. Deng "Dynamic Secure Cloud Storage with Provenance", pp. 442–464, Springer-Verlag Berlin Heidelberg 2012

[5]. Gansen Zhao, Chunming Rong, Jin Li, Feng Zhang, and Yong Tang " Trusted Data Sharing over Untrusted Cloud Storage Provider", IEEE second International Conference on Cloud Computing Technology and Science (CloudCom), 2010

[6]. Piotr K. Tysowski and M. Anwarul Hasan "Re-Encryption-Based Key Management Towards Secure and Scalable Mobile Applications in Clouds" , in IACR Cryptology ePrint Archive, 2011, pp. 668

[7]. Mr. Ashwin Chandra C and Ms. Dharani S "Decentralized access control with Aggregate Key Encryption for Data Stored in Cloud", International Journal of Innovative Research in Computer and Communication Engineering Vol.2, Special Issue 1, March 2014

[8]. M.J. Atallah, M. Blanton, N. Fazio, and K.B. Frikken, “Dynamic and Efficient Key Management for Access Hierarchies,” ACM Trans. Information and System Security, Vol. 12, no. 3, pp. 18:1-18:43, 2009

[9]. J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, “Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records,” Proc. of ACM Workshop Cloud Computing Security (CCSW ’09),pp. 103-114, 2009

[10]. V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data,” Proc. of 13th ACM Conference On Computer and Comm. Security (CCS ’06), pp. 89-98, 2006

[11]. http://www.docstoc.com/docs/99263024/A-Survey-of-Identity-Based-Cryptography

[12]. Bhavani Thuraisingham, Vaibhav Khadilkar, Anuj Gupta, Murat Kantarcioglu and Latifur Khan " Secure Data Storage and Retrieval in the Cloud", Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2010 6th International Conference.

[13]. Cheng-Kang Chu, Sherman S.M. Chow, Wen-Guey Tzeng, Jianying Zhou, and Robert H. Deng "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage", IEEE transactions on parallel and distributed systems, vol. 25, no. 2, 2014