International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-11, September 2019

Key Aggregate Cryptosystem in Data Sharing in Cloud Performance

Aishwarya Seth, Bhoomi Gupta

Abstract— Sharing data using cloud storage has become a prominent part of a lives therefore, it is necessary to be able to share data to the people we choose and which data we choose. In this article, Key Aggregate Cryptosystem scheme, a public key encryption scheme which produces fixed-size cipher texts [1], is studied comprehensively along with its application in sharing data by users across cloud platform. KAC provides efficient and flexible delegation of keys for sharing selective data using single aggregated key without compromising on other data stored in cloud. The scheme is implemented using python file integrated in the backend of a website, which is in Django and the cloud storage used is Amazon S3. The data is stored in buckets and can be accessed using the website. Users can also upload and download their data from the website, or even share after uploading on the cloud.

Keywords: KAC, Cloud storage, Data sharing, Key aggregation, Public key cryptosystem

I. INTRODUCTION

Sharing data has always been one of the most commonplace tasks we do. Though with time, methods employed for doing so have kept on changing. Recently, the trend of people preferring to store their data on cloud, especially organizations that have to store gigantic amounts of data, has become popular. The reasons for this are diverse in cognizance of various advantages like reduced costs of storage, accessibility from anywhere and easy backup and recovery. However, sharing data via cloud is not secure in itself and needs some form of encryption, particularly for sharing sensitive data. There have been many studies conducted and algorithms proposed for secure sharing of data and many prominent contributions have been made till date. One study or algorithm is of notable interest viz., Key Aggregate Cryptosystem [1]. KAC is a cryptographic algorithm proposed by Cheng-Kang Chu Et. Al., which does not only encrypt data for satisfying security concerns but also provides flexible delegation rights. Therefore, it solves the problem of secure data sharing without any compromise on efficiency and flexibility of data sharing. The merits of KAC can be cited by following illustration(s).

Manuscript published on 30 September 2019.
***Correspondence Author(s)**
Aishwarya Seth, Student, MAIT, Delhi, India.
(E-mail: sethaishwarya1997@gmail.com)
Dr. Bhoomi Gupta, Assistant Proffessor, MAIT, Delhi, India.
(E-mail: guptabhoomi@gmail.com)

An organization stores its data on cloud and, leakage of sensitive data is a major issue for every organization. Therefore, the organization will want to share its confidential data with its employees such that each employee can access only that data with which that employee is concerned. Another scenario can be that Alice wants to share some of the photos she has stored on cloud with Bob, without divulging the contents of rest of the photos to Bob. Both of these problems are in correlation with real life scenarios and, in fact are pretty conventional. Such type of problems can be overcome by KAC as it uses flexible hierarchy for delegation of access rights.

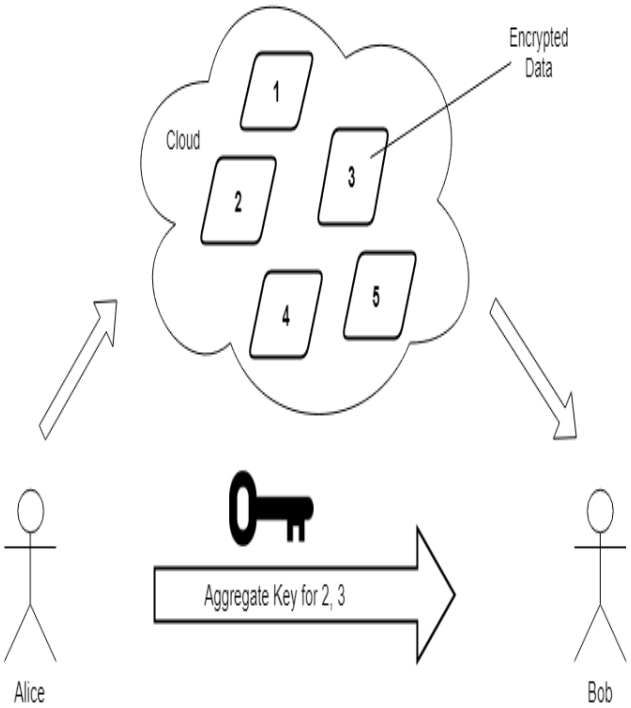

Figure 1: Alice sharing files with Bob via cloud using aggregate key

II. RELATED WORKS

There have been several studies conducted and various solutions presented for solving the problem of flexible delegation of access rights. The problem was initially sought to be solved by fixed tree hierarchies in which the data was predefined in a tree structure and the keys were generated according to the levels. As a result, parent node can be used to derive child nodes.

Retrieval Number: K25710981119/19©BEIESP
DOI: 10.35940/ijitee.K2571.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

The solutions proposed were symmetric key systems which aimed at minimizing the expense of storage [2] [3] [4]. However, these solutions can be generalized in the form of graphs as well [5] [6].An example is illustrated to explain the same. Bob wants to share some selected folders with Alice in his projects directory which was at say *N* depth. In order to do this, he will have to generate *K* keys for *K* folders to be shared because sharing projects directory with Alice would expose all the other data in that directory as well. As the directories diversify and the directory structure grows complex numerous keys might need to be generated for individual nodes which can become cumbersome and unmanageable. Thus, complex hierarchies are not benefited by this scheme. Hence, an encryption method is needed which is independent of pre-defined hierarchies and support flexible delegation.

Compact key was used for initially solving the problem of flexible delegation in symmetric key encryption [7]. These solutions mostly comprised of tree or cyclic graph structures and keys generated were symmetric in nature, that is, same key was used for encryption and decryption. The schemes have tried reduction of key size but sharing has not caused any problems. However, since keys are symmetric in nature and there is obscurity in as to how to apply the schemes in public key settings therefore, they are not feasible in many applications. Similar approach was undertaken in Identity-based encryption, which is basically a form of public key cryptosystem. In this, any form of identification of users can be used to serve as public key. Aggregation of keys were done using identities but they all had to be from different divisions [8] [9] [10].

Collusion resistant broadcast encryption scheme developed by Professor Dan Boneh et.al. [11] has been fundamental to development of Key-Aggregate Cryptosystem. It is a public key encryption scheme designed for preventing collusion. The scheme had setup, encrypts and decrypts steps in the broadcast encryption algorithm and gave fixed size private keys and cipher texts. Nonetheless, it was applicable only for small number of group elements, two to be precise. Although another generalized scheme was proposed for further research in the study.

III. KAC

As stated before, KAC scheme has been motivated by study by Professor Dan Boneh et. al. [11] on Collusion resistant broadcast encryption scheme, a public key encryption scheme for broadcast scenario for prevention of collusion but in this system, key decrypts data for a particular index only. The algorithm for KAC can be stated as follows:

- The first step is for an account to be **setup** by the sender on an untrusted server by giving number of cipher text classes say n, and a security level parameter as input,

thereby bounding the class index to be in the range from 1 to n. A public system parameter say param, will be generated as on output and will be given as an input into all of the following steps of the algorithm.

- Second step is for the sender to **generate** a random pair of **keys**, public key, pk and master-secret key, msk.
- Third step is for **encryption** of data and is executed by any user wishing their data to be encrypted. In this, the desired message, the public key and index of the ciphertext class are given as input and a ciphertext is generated as an output.
- Fourth step includes a set of indices of different ciphertext classes and master secret key being given as inputs and aggregate key, KS of the set being **extracted** as the output. This operation is done in order to be able to delegate the capability of decryption to the receiver for a certain of set of ciphertext classes.
- The final step is to perform **decryption** at receiver's end by taking the aggregate key, set of indices of ciphertext classes, the index of ciphertext classes wished to be accessed and the ciphertext as input.

In this algorithm system parameters can be computed beforehand thereby, reducing our time further. This algorithm can be integrated into backend and used for encryption and decryption for a fixed amount of ciphertext classes. The storage service on cloud being used is Amazon S3 and backend is made using Django in this study for implementation of KAC.

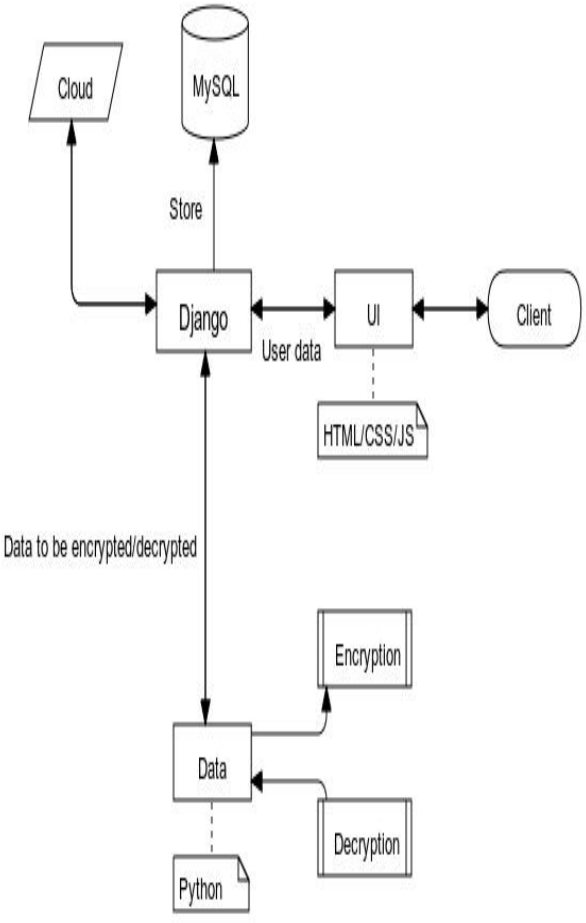

Figure 2. System architecture

Retrieval Number: K25710981119/19©BEIESP
DOI: 10.35940/ijitee.K2571.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

IV. EXPERIMENTAL ANALYSIS & RESULTS

r	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Setup	4.0								
Extract	1	2	3	4	4	5	5	5	6
Decrypt	2	4	6	7	8	10	10	11	12

Table 1: Performance of KAC for h = 8 with respect to different delegation ratio r (in milliseconds)

Here, test machine is a Dell G3 15 system with eight cores running Windows 10, with 12 GB RAM. An optimization can be made by reducing the computation time for exponentiation of g in the Setup algorithm by using PBC consisting of a preprocessing function, thereby making it more efficient. Table 1 concludes the observations made in the experiment. The time of execution of first three steps of algorithm are independent of the delegation ratio r but, for fourth and fifth of algorithm, that is, extraction step and decryption step, it increases in a linear fashion against delegation ratio r. Another key observation to be noted from conclusions is that the execution time of Decrypt is approximately twice of running time of Extract.

V. CONCLUSION AND FUTURE WORK

An efficient encryption system with the power of single aggregate key is adopted in this implementation. This ensures confidentiality for scalable data sharing in cloud. Thus, Key-Aggregate Cryptosystem (KAC) [1], a public-key encryption scheme helps in maintaining data privacy and security. With the help of KAC system, users can share their data over cloud partially using fixed size key pair of public key and master key and also, using a lone fixed size aggregate key, they can decrypt their data. The major drawback of this scheme is that the amount of maximum ciphertext classes has to be defined beforehand therefore, enough cipher text classes are needed to be reserved for future extension or, the public-key can be expanded as well. But the amount of cipher texts usually increases quickly in cloud storage therefore; more optimal solution can be achieved by making the size of parameter unconstrained and free from the maximum amount of cipher text classes. Also, the keys delegated should be handled sensitively and kept safely as they are prone to leakage. This can be solved by designing a leakage resilient scheme which facilitates efficient and flexible key delegation.

REFERENCES

1. S.S.M. Chow, J. Zhou, W.G. Tzeng, C.K. Chu, and R.H. Deng, "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 2, pp. 468-477, Feb. 2014.
2. P. D. Taylor and S. G. Akl, "Cryptographic Solution to a Problem of Access Control in a Hierarchy," ACM Transactions on Computer Systems (TOCS), vol. 1, no. 3, pp. 239–248, 1983.
3. W.-G. Tzeng, "A Time-Bound Cryptographic Key Assignment Scheme for Access Control in a Hierarchy," IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 14, no. 1, pp. 182–188, 2002.
4. S. E. Tavares and G. C. Chick, "Flexible Access Control with Master Keys," in Proceedings of Advances in Cryptology - CRYPTO '89, ser. LNCS, vol. 435. Springer, 1989, pp. 316–322.
5. K.J.R. Liu and Y. Sun, "Scalable Hierarchical Access Control in Secure Group Communications," Proc. IEEE INFOCOM '04, Hong Kong, China, 2004, pp. 1296-1306.
6. R. S. Sandhu, "Cryptographic Implementation of a Tree Hierarchy for Access Control," Information Processing Letters, vol. 27, no. 2, pp. 95–98, 1988.
7. J. Benaloh, "Key Compression and Its Application to Digital Fingerprinting," Microsoft Research, Tech. Rep., 2009.
8. Z. Chen, F. Guo, and Y. Mu, "Identity-Based Encryption: How to Decrypt Multiple Cipher texts Using a Single Decryption Key," in Proceedings of Pairing-Based Cryptography (Pairing '07), ser. LNCS, vol. 4575. Springer, 2007, pp. 392–406.
9. M. K. Franklin and D. Boneh, "Identity-Based Encryption from the Weil Pairing," in Proceedings of Advances in Cryptology - CRYPTO '01, ser. LNCS, vol. 2139. Springer, 2001, pp. 213–229.
10. S. S. M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, "Practical Leakage-Resilient Identity-Based Encryption from Simple Assumptions," in ACM Conference on Computer and Communications Security, 2010, pp. 152–161.
11. D. Boneh, B. Waters, and C. Gentry, "Collusion Resistant Broadcast Encryption with Short Cipher texts and Private Keys," in Proceedings of Advances in Cryptology - CRYPTO '05, ser. LNCS, vol. 3621. Springer, 2005, pp. 258–275.