IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834, p-ISSN: 2278-8735
PP 19-24
www.iosrjournals.org

Key-Aggregate Cryptosystemfor Scalable Data Sharing in Cloud Storage

Shinde .V.[1], Kurhade Vidya[2], Boraste Sujata[3],Gaikwad Prajakta[4] & Baviskar Shweta[5]
[1,2,3,4,5](Comp Engg. Dept.,GNS COE Nasik,SPPUniv.Pune(MS), India)

Abstract *:Sharing of Data is an main functionality in cloud Computing. In this system, we show how to share data in securely, flexibly, efficiently, and with others in cloud computing. We design new public-key cryptosystems which produce Fixed-size encrypted texts such that efficient delegation of decryption rights for any set of Encrypted data are possible. The innovation is that one can aggregate any set of secret keys and make them as compressed as a single key, but on all sides of the power of all the keys being aggregated. In other words, the secret key holder can release a fixed-size aggregate key for flexible choices of encrypted text set in cloud storage space, but the other ciphered files outside the set remain secret. This compress aggregate key can be conveniently sent to others or be stored in a E-mail with very limited secure storage space. We provide formal security analysis of our system in the standard model. We also describe other application of our schemes.*

Keywords*- Cloud Storage, Data Sharing, Key-Aggregate Encryption.*

1. INTRODUCTION

Storing data on cloud is gaining popularity recently. In enterprise, we see the increase in demand for data outsourcing, which assists in the planned management of business data. It is also used as a basic technology behind many online services for personal applications. Now, it is easy to apply for free accounts for email, photo album, file sharing and/or remote access, with storage size more than 25GB.Together with the current wireless technology, users can retrieve almost all of their files and emails by a cell phone in any side of the world. data Confidentiality, a traditional way to ensure it is to rely on the server to enforce the access control after authentication, which means any unpredicted privilege rise will expose all data. Data from different clients can be present on separate virtual machines but reside on a single physical machine. Data in a destination VM could be stolen by instantiating another VM co-resident with the destination one. Regarding availability of files, there are a number of cryptographic schemes which go as far as allowing a third-person auditor to check the availability of files on behalf of the sender without leaking anything about the data, or without compromising the data owners secrecy. Likewise, cloud users possibly will not hold the strong conviction that the cloud server is doing a good job in terms of secrecy. A cryptographic solution, with Stated security relied on number-theoretic assumptions is more attractive whenever the useris not perfectly happy with trusting the security of the Virtual Machine or the honesty of the technical member. These users are encouraged to encrypt their files with their own keys before uploading them on to the cloud. Sharing of data is an vital functionality in cloud storage.

The demanding problem is how to effectively share cipher text. So the users can download the cipher text from the storage, decrypt them, then upload them on to the cloud for sharing, but it loses the value of cloud computing. Users should be able to hand over the access rights of the sharing data to others so that they can access these data from the cloud directly. Below we will take Dropbox as an example. Assume that 'A' puts all her private data on Dropbox, and she does not want to leak her photos to everyone. Due to various data leakage possibility A cannot feel comfortable by just relying on the privacy protection mechanisms provided by Dropbox, so she encrypts all the data using her own keys before uploading. Now, A's friend, B, told her to share the data taken over all these years which B appeared in. A can then use the share function of Dropbox, but the problem is how to hand over the decryption rights for these data to B. The possible option A can choose is to firmly send B the secret keys involved. obviously, there are two great ways for her under the traditional encryption concept:

A encrypts all files with a one encryption key and gives B the corresponding private key directly.

- A encrypts files with Different keys and sends B the corresponding private keys.

clearly, the first option is not efficient since all unselected data may be also leaked to B. For the second option, there are practical concerns on efficiency. The number of such keys is as many as the number of the

shared data, say, a Hundred. Transferring these private keys naturally requires a secure medium, and storing these keys requires rather costly secure storage. The costs and complexities involved generally boost with the number of the decryption keys to be shared. In short, it is very deep and costly to do that.

Encryption keys also come with two flavors — symmetric key or asymmetric (public) key. Using symmetric encryption, when A wants the data to be originated from a third person, she has to give the encryptorof her key; clearly, this is not always desirable. By dissimilarity, the encryption and decryption key are different in public-key encryption. The use of public-key encryption gives more flexibility for our system. For example in business settings, every employee can upload ciphered data on the cloud storage server without the knowledge of the company's secret key. Therefore, the best solution for the above problem is that A encrypts files with different public-keys, butonly sends B a one decryption key. Since the decryption key should be sent via a secure medium and kept secret, small key size is always useful. Especially, these secret keys are usually stored in the tamper-proof memory, which is relatively costly. The present research hard work mainly focus on minimizing the communication requirements (such as bandwidth, rounds of communication) like aggregate signature However, not much has been done about the key itself.

Our Contributions

In modern cryptography, a fundamental problem we often study is about leveraging the secrecy of a small piece of knowledge into the ability to perform cryptographic functions (e.g. encryption, authentication) multiple times. In this system, we study how to make a decryption key more powerful in the sense that it allows decryption of multiple Encrypted data, without increasing its size. A shares files with identifiers 2, 3, 6 and 8 with B by sending him a one aggregate key. "To design an capable public-key encryption system which supports flexible delegation in the sense that any subset of the encrypted data (produced by the encryption scheme) is decryptable by a fixed-size decryption key." We find the solution to this problem by designing a special type of public-key encryption which we call key-aggregate cryptosystem (KAC). In KAC, users encrypt a message not only under a public-key, but also under an identifier of ciphertext called class. That means the encrypted are further categorized into different classes. The key owner holds a secret key, which can be used to extract secret keys for different classes. More Significant, the extracted key have can be an aggregate key which is as compress as a secret key for a one class, but aggregates the power of lots of such keys, i.e., the decryption power for any subset of encrypted classes. With our solution, A can simply send B a one aggregate key via a secure e-mail. B can download the encrypted files from A's Dropbox storage space and then use this aggregate key to decrypt these encrypted files. The sizes of encrypted data, public-key,secret key and aggregate key in our KAC schemes are all of fixed size. The public system parameter has size linear in the number of ciphertext, but only a small part of it is needed each time and it can be fetched on demand from large cloud storage.

2. LITERATURE SURVEY

This section we compare our basic key aggregate cryptosystem scheme with other possible solutions on sharing in secure cloud storage.

Cryptographic Keys for a Predefined Hierarchy

Cryptographic key assignment schemes (e.g., [1], [2], [3], [4]) goal to reduce the cost in storing and managing secret keys for general cryptographic use. Using a tree structure, a key for a given branch can be used to get the keys of its descendant nodes. Just permitting the parent key implicitly grants all the keys of its descendant nodes. Sandhu et.al [5] proposed a method to generate a tree hierarchy of symmetric keys by using frequent evaluations of pseudo random function or block-cipher on a fixed secret. The method can be generalized from a tree to a graph. Advanced cryptographic key assignment concept support access policy that can be modeled by an connected graph or a disconnected graph [6], [7], [8]. Most of these concept produce keys for symmetric-key cryptosystems, even if the key derivations may need modular arithmetic as used in public-key cryptosystems, which are generally more costly than "symmetric-key operations" such as pseudo random function We Consider the tree structure as an example. A can first classify the ciphertext classes according to their subjects.

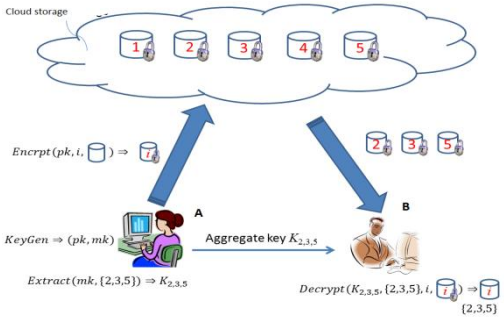
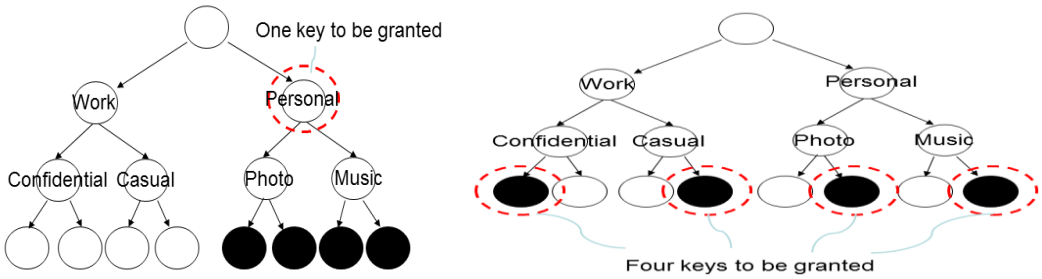


Fig. 1: Using KAC for data sharing in cloud storage

Each node in the tree represents a secret key, while the child nodes represents the keys for single ciphertext classes. Filled circles represent the keys for the classes to be delegated and circles circumvented by dotted lines represent the keys to be permitted. Note every key of the non-child node can derive the keys of its descendant nodes.



(a) (b)

Fig 2: Network

Fig:Compact key is not always possible for a fixed hierarchy In Figure (a), if A wants to share all the files in the "personal" category, she only needs to permit the key for the node "personal", which automatically permit the delegatee the keys of all the below nodes ("photo", "music"). This is the perfect case, where more classes to be shared belong to the same branch and thus a parent key of them is enough. But, it is still difficult for general cases. As shown in Figure(b), if A shares her demo music at work ("work"!"casual"!"demo" and "work"!"confidential"!"demo") with a co-worker who also has the privileges to see some of her personal files, what she can do is to give more keys, which leads to an rise Fig:Compact key is not always possible for a fixed hierarchy In Figure (a), if A wants to share all the files in the "personal" category, she only needs to permit the key for the node "personal", which automatically permit the delegatee the keys of all the below nodes ("photo", "music").

This is the perfect case, where more classes to be shared belong to the same branch and thus a parent key of them is enough. But, it is still difficult for general cases. As shown in Figure(b), if A shares her demo music at work ("work"!"casual"!"demo"and"work"!"confidential"!"demo") with a co-worker who also has the privileges to see some of her personal files, what she can do is to give more keys, which leads to an rise in the total key size. One can see that this method is not flexible when the classifications are more complicated and she wants to share different sets of data to different person. For this delegatee in our example, the number of permitted secret keys becomes the same as the number of classes. In general, tree structure approaches can solve the problem incompletely if one intends to share all data under a certain branch in the tree. On an average, the number of keys rise with the number of branches. It is unlikely to come up with a hierarchy that can save the number of total keys to be permitted for all simultaneously.

Compact Key in Symmetric-Key Encryption

The same problem of supporting flexible hierarchy in decryption delegation. Benaloh et al. [8] describe an encryption scheme which is originally designed for concisely transmitting large number of keys in multicast scenario [9]. The development is simple and we briefly assess its key derivation process here for a concrete description of what are the required properties we want to achieve.

This approach achieves same properties and performances as our system.But, it is designed for the symmetric-key setting instead. The sender needs to get the corresponding secret keys to encrypt files, which is not proper for many applications. Since their method is used to generate a secret value rather than a pair of public or secret keys, it is uncertain how to apply this idea for public-key encryption scheme. Finally, there are schemes which try to minimize the key size for achieving authentication in symmetric-key encryption. However, sharing of decryption power is not a concern in these schemes.

Compact Key in Identity-Based Encryption

Identity-based encryption (IBE) (e.g., [10]) is a type of public-key encryption in which the public-key of a user can be set as an identity-string of the user .There is a trusted party called private key generator (PKG) in IBE which holds a master-secret key and issues a secret key to every user with respect to the user identity. The sender can take the public parameter and a user identity to encrypt a message. The recipient can decrypt this ciphertext by his secret key. Guo et al. [11] tried to build IBE with key aggregation. One of their system [11] assumes random oracles but another does not. In their system, key aggregation is constrained in the sense that all keys to be aggregated must come from another "identity divisions". While there are an exponential number of identities and thus secret keys, only a polynomial number of them can be aggregated. Firstly, their key-aggregation [11]comes at the expense of O(n) sizes for both ciphertexts and the public parameter, where n is the number of secret keys which can be aggregated into a constant size one. This greatly rise in the costs of storing and transmitting ciphertexts, which is unpractical in many conditions such as shared cloud storage. Another way to do this is to apply hash function to the string denoting the class, and keep hashing repeatedly until a prime is obtained as the output of the hash function.

They mentioned, our schemes feature constant ciphertext size, and their security holds in the standard model. In fuzzy IBE one single compact secret key can decrypt ciphertexts encrypted under many identities which are close in a certain metric space, but not for an arbitrary set of identities and therefore it does not match with our idea of key aggregation.

3. PROPOSED SYSTEM

IN OUR PROPOSE SYSTEM

We find the solution changes the way We keep the keys at sender side and how the keys will be receive at receiver side.

Key-aggregate encryption

We describe how to use key aggregate cryptosystem In a consequence of its application in cloud storage.

Framework

A key-aggregate data owner institutes the public system parameter Via Setup and generates a secret3 key encryption scheme consists of five polynomial-time algorithms as follows. The data owner institutes the public system parameter Via Setup and generates a secret3 key pair via Key Generation. Messages can be ciphered via Encrypt to anyone who also decides what Encrypted class is associated with the readable data message to be encrypted. The data owner can use the secret key to generate an aggregate decryption key for a set of Encrypted classes is Extracted. The generated keys can be passed to delegates securely to secure e-mails or secure devices.o9ol

- Setup1(1, n): Perform by the data owner to setup an Account on an untrusted server. On input a security level parameter 1,n and the number of Encrypted classes n and it out puts of the public system parameter param, which is mislaid from the input of the other algorithms for succinctness.
- Key Generation: Perform by the data owner to differently Generate a secret key pair.
- Encrypt : perform by anyone who wants to Changeable to unreadable form.
- Extract(msk; S):perform by the data owner for allocating the decrypting authority for a certain set of ciphertext classes to a agent. On input the secret key msk and a set S of indices equivalent to different classes, it outputs to the aggregate key for set S denoted by KS.
- Decrypt(KS; S; i; C):perform by a agent who give an aggregate key KS generated by Extract. On input KS, the set S, an index i denoting the Encrypted class to the ciphertext C belongs to, and C, it outputs the decrypted result m if i 2 S.

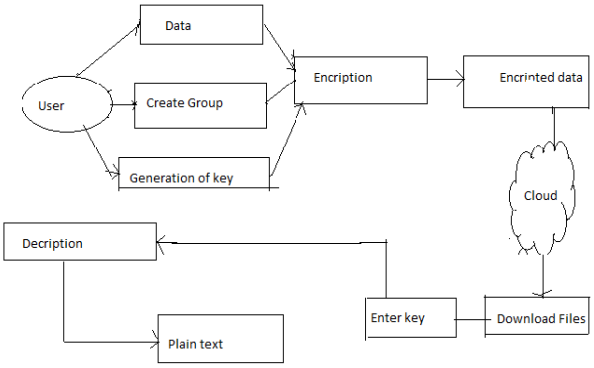


Fig. System Architecture

Sharing Ciphertext Data

A official application of key aggregation cryptosystem is data sharing. The key aggregation term is especially useful for the delegation can be efficient and flexible. The Schemes enable a content provider to share her data in a personal way, with a fixed and small ciphertext development, by dispensing to each valid user a single and small aggregate key. we design the main feature of sharing of data in cloud storage using key aggregation cryptosystem. Suppose A wants to share her data to m1;m2; : : : ;m on the server. She first performs Setup(1_; n) to get param and perform Key Generation to get the secret key pair. The system parameter param and public-key pk can be made public and private key msk should be kept secret by A.

The goal of cloud computing is to apply traditional supercomputing or high performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per sec, in consumer oriented applications such as financial port folios, to deliver personalized information, to provide data storage or to power large immersive computer games. The user can upload the data on cloud then user can create a group select the files in those group an encrypt those files then upload those data on cloud then receiver login to cloud for uploading the data entering the keys then data has decrypted.

Algorithms for Secrete key encryption
1.start
2.select file which we want to send.
3.suppose files are f1,f2,f3,generate random key of respective file.
suppose that are k1,k2,k3
Let,k1=123,k2=456,k3=789
4.combine the keys and add separator(ie.0)between them
5.Take one Quadratic equation,
$F(x)=n1x+n2x^2+s$
Let,n1=19,n2=6,x=3(as sending files are 3)
6.$F(x)=(19*3)+6*(3^2)+s$
$F(x)=57+54+s$
$F(x)=111+s$
7.$f(x)=111+12304560789$
$=1.230456e^{10}$
8.$1.230456e^{10}$ this is aggregate key send via email.

4. CONCLUSION AND FUTURE SCOPE

This application is useful in cloud computing to scalable system using different cryptographic algorithm. How to defend users data privacy is a central problem of cloud storage. With more mathematical concept, cryptographic schemes are getting more multipurpose and often involve multiple keys for a single system. In this System we consider how to "compact" secret keys in public-key cryptosystems which support delegation of secret keys for different cipher text classes in cloud storage. Our approach is more flexible than hierarchical key

assignment which can only save storage spaces if all key-holders share a similar set of privileges. A limitation in our work is the predefined bound of the number of maximum cipher text classes. In cloud storage, the number of cipher texts usually grows rapidly. So we have to reserve enough cipher text classes for the future extension.

ACKNOWLEDGEMENTS

I express my gratitude to all anonymous researchers for providing us such helpful opinion, findings, conclusions and recommendations.

REFERENCES

[1] S. G. Akl and P. D. Taylor, *"Cryptographic Solution to a Problemb of Access Control in aHierarchy,"* ACM Transactions on Computer Systems (TOCS), vol. 1, no. 3, pp. 239– 248, 1983
[2] G. C. Chick and S. E. Tavares, *"Flexible Access Control with Master Keys,"* inProceedings of Advances in Cryptology – CRYPTO '89, ser. LNCS, vol. 435. Springer,1989, pp. 316–322.
[3] W.-G. Tzeng, *"A Time-Bound Cryptographic Key Assignment Scheme for Access Control in a Hierarchy,"* IEEE Transactions onKnowledge and Data Engineering(TKDE), vol. 14, no. 1, pp. 182–188, 2002.
[4] G. Ateniese, A. D. Santis, A. L. Ferrara, and B. Masucci, *"Provably-Secure Time-BoundHierarchical Key Assignment Schemes,"* J. Cryptology, vol. 25, no. 2, pp. 243–270, 2012.
[5] R. S. Sandhu, *"Cryptographic Implementation of a Tree Hierarchy for Access Control,"* Information Processing Letters, vol. 27, no. 2, pp.95–98, 1988.
[6] Y. Sun and K. J. R. Liu, "*Scalable Hierarchical Access Control in Secure GroupCommunications*," in Proceedings of the 23th IEEE International Conference on Computer Communications (INFOCOM '04). IEEE, 2004.
[7] Q. Zhang and Y. Wang, *"A Centralized Key Management Schem for Hierarchical Access Control,"* in Proceedings of IEEE GlobalTelecommunication Conference (GLOBECOM'04). IEEE, 2004, pp. 2067–2071.
[8] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, *Patient Controlled Encryption:Ensuring Privacy of Electronic MedicalRecords,"* in Proceedings of ACM Workshop on Cloud Computing Security (CCSW '09). ACM, 2009, pp. 103–114.
[9] J. Benaloh, *"Key Compression and Its Application to Digital Fingerprinting,"*Microsoft Research, Tech. Rep., 2009.
[10] D. Boneh and M. K. Franklin, "*Identity-Based Encryption from the Weil Pairing,"* in
[11] Proceedings of Advances in Cryptology – CRYPTO '01, ser. LNCS, vol. 2139. Springer, 2001, pp. 213–229.
[12] F. Guo, Y. Mu, and Z. Chen, *"Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key*," inProceedings of Pairing-BasedCryptography (Pairing '07), ser. LNCS, vol.4575. Springer, 2007, pp. 392–406.