IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

Efficient Framework for Secure Data Sharing With Key Aggregate Cryptosystem

Pallavi Mathur[1], Nachiket Mahamuni[2], Nikhil Gondane[3], Kaushal Singh[4]

Computer Department, DYPIET, Pune University (UoP), Pune, India [1,2,3,4]

Abstract: Cloud storage in simple terms is nothing but storing data on-line in cloud that can be accessed from multiple and connected resources. Using cloud to store the data is a very practical thought as we can share our information in trustworthy domain, with great efficiency, Besides that it provides some other functionalities in terms of disaster recovery, smart accessibility etc. New public–key cryptography is presented that came out to be Key aggregate cryptosystem (KAC), in which constant size ciphertexts are produced in order to make the task of rendering the decryption rights feasible for produced mass of ciphertexts. KAC comprises the thought of integrating different keys generated for different sets of ciphertext which yields a single aggregate key accommodating power of all the keys. After which a single key can be released to someone in trusted domain through a secure channel and using this key one can decrypt set of allowed files only, others remaining unaddressed.

Keywords: Cloud storage, Data sharing, Ciphertext, Key-aggregate, Encryption.

I. INTRODUCTION

Cloud storage has gained tremendous popularity now-a-days of course due to functionality it provides. It stands to gain advantages over traditional ways of storing the information/data. Cloud storage is storing our personal, casual, or work related data off-site to the physical storage that is maintained by third party. Cloud storage is saving of digital information in logical pool and physical storage carrying loads on multiple servers that are manage by third party. Third party is liable for keeping, storing and maintaining information on the market. And accessible and physical atmosphere ought to be protected in runtime running in the slightest degree time. Instead of storing information to the disk drive or the other native storage, we tend to save information to remotely located storage devices which is accessible from anyplace and at any given point in time. It keeps from carrying physical devices everywhere you go. By exploitation cloud storage we can access data from any pc over the net that omitted limitation of accessing data from same pc where it's kept. While considering information (content) privacy and protection, resolution is to encipher information before uploading to the server with user's own key. And survey has shown that it is always a good practice to encrypt the personal information before uploading on a server as there could be many backdoors through which an intruder may hack into your data stored online. To understand let us assume an instance, organization might grant access rights to go and roam through a part of sensitive information to their staff. However difficult task is to share enciphered information securely. Traditional tactic is user will transfer the encrypted information from storage, decipher that information and send it to share with others; however it loses the importance of storing data on a cloud.

Cryptography technique is applied in a very 2 major ways- one is even key cryptography and otherone is uneven key encryption.

In even key cryptography, similar keys are used for cryptography and coding i.e. same key to encrypt and decrypt the knowledge. Against this, in uneven key encryption totally different keys are used, public key for cryptography and personal key for coding i.e. different keys for converting message to cipher and cipher to message back in its original format.

Suppose Alice puts her all information on demo.com and she does not need to show her information to everybody. Thanks to information outpouring prospects she does not trust on privacy mechanism provided by demo.com, thus she encipher all information data before she uploads it on the server. If Bob ask her to share some information with him then Alice will encipher her files and will generate number of keys equals to no of files/docs. Its okay to employ this scheme for few no. of docs but suppose there are thousands of files then that many no of keys are to be produced. But using this way is very unrealistic as it is surely not feasible for her to handle this many no of keys. Besides this it needs to have more secure channel to hand over this many no. of keys. It increases cost of using resources and complexity. So there is a way to surmount this hurdle. As Alice encrypts no of files, secrete keys that are generated can be fused together to form an aggregate-key. This aggregate master key has the power of all the keys and can be used to decrypt multiple enciphered files at a time. This reduces cost of resources, no need to have secure channel to share the keys as there is going to be a single key to be deployed to.

II. RELATED WORK

In [2], authors states in electronic health records ,medical related services and health related service, providers uploads and updates patients record and when so ever they need to access their information or health related

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

documents and other health data they can at any time. Besides that they can render their rights to decrypt the data to a guardian or friend so that they can access, retrieve, delete, and modify their records. Here they say a particular patient can generate his/her own decryption keys. But this altogether sounds somewhat risky and problematic as if access is granted then any one who has decryption rights (decryption key generated by patient himself) can delete or replace the information without having the idea of what type of file is getting modified. Hence one has to maintain different keys for different files in order to avoid this scenario. Hence no of keys will have to generate and that many no of keys will have to be deployed.

M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, in "Dynamic and Efficient Key Management for Access Hierarchies," ACM Transactions on Information and System Security; proposed tree based mechanism where classes are set in hierarchy according to its order. In this if a person gets access rights to access a particular class in hierarchy then he access the attributes of that particular class also he can access its descendant classes as well. Here the problem is spurt may be seen in no of keys as the branch in hierarchy grows.

In [3], authors proposed: attribute based fine-grained access control as a way of encryption. Generally cryptographic methods are employed when there is need of sharing of important and sensitive information over the net. If one is to be given access rights then he must carry a private key in order to decrypt the data and hence the data owner has to release his private key this schema is known as coarse grained. This article proposed fine-grained schema with ABE. i.e. each set of message will have some attributes on which a private key can be mapped, which in turn restricts third party user from decrypting the information.
The scheme proposed by the authors in [11], states: trustworthy key generator in IBE holds the master secrete key and discloses secrete keys to a particular user based on identity measures. Employing key-aggregation yields a system where encryptor accepts simple public parameter and user identity in order to cipher the data and a receiver will use his own secrete key to decrypt the ciphertext-set. But this may increase resource related expenses as storing and deploying secrete keys becomes expensive.

Here in [4], a private key can be used to map multiple identities i.e. public keys. That means using various public keys one can encrypt multiple files of information and a private key is used to decrypt multiple cipher messages that can map multiple public keys encompassed with it. This schema increases efficiency and reduces efforts of producing multiple keys for multiple documents of information.

III. KEY-AGGREGATE ENCRYPTION

Following fig. 1 gives idea about how data can be shared using key-aggregate encryption.

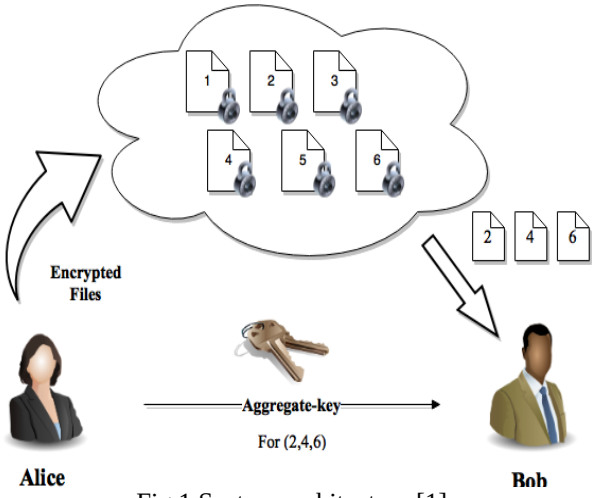

Fig 1 System architecture [1]

Supreme goal of KAC is to securely share the data. The key aggregation is an important aspect particularly when we expect the task of rendering access rights to be economical and versatile. In KAC, one can share the knowledge data content in a very confidential and selective means, with a set of enciphered text, by distributing to every approved and authenticated user one fusedkey.

Data sharing in KAC scheme, illustrated in Figure one. Suppose Alice puts her knowledge m1, m2,...., mn on the server and now she is willing to share her data. She initially opt to perform Setup() to induce set up connection. After the secrete key is induced, the message 'm' is converted into cipher message. The enciphered knowledge-data measure uploaded to the server. Alice will update/maintain Alice's knowledge content on the server. Once Alice(data owner) is willing to share a collection S of her knowledge to someone, say Bob, she will make the aggregate key for Bob. once it is ready it can be handed over to Bob through secure channel after which he can transfer the document contents he is allowed to go through. That is, every time index ϵ set of allowed indices, Bob is allowed to downloads and decipher the knowledge content from the server.

Whomsoever is supposed to make use of this system will first have to create his account and register himself either as a data owner or a user. According to his current role in the system he can avail various associated privileges to that role.

IV. IMPLEMENTATION IDEOLOGY

Modules related to these schemes could be:

Module1: (Data owner side)
Data owner who is willing to upload, update and share knowledge set will create his account first and authenticate himself by user_id or user_name and password. i.e. Setup.
Now the particular person is in the role of data owner, and hence he can avail services like file upload, view file, update file etc.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

Besides that he will encrypt the data and will have to generate a key for the same to encipher(KeyGen). At the time he is requested to share his data instead of no of keys he will make an mixture key(agg.key) and deploy the same through some secure medium like email. (Extract)

Module2: (User side)
User is one how is requesting data owner to share his knowledge set, now as the particular person is in user role, he will have to create an account and authenticate himself first(using user_name and password).
He will use the received agg.key and the set of indices to decrypt the files he is allowed to go through. Likewise he will decipher the data set and can download the content.
Here both i.e. data owner and user are nothing but only the roles. And these roles are purely interchangeable ,a data owner can be a user at times and a user can be a data owner when he is generating an agg.key.

The phases are as follows:

- **Setup:** performed by the data owner to perform setup
- **KeyGen:** performed by data owner to induce the key
- **Encrypt:** by data owner to encipher knowledge content
- **Extract:** performed to induce master key
- **Decrypt:** performed by the user to decipher allowed set of knowledge-content.

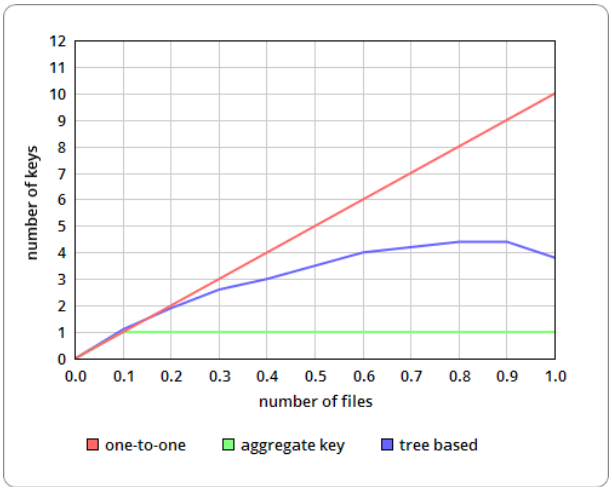

Fig. 2 Graph (no.file v/s no of keys)

Here in above graph shows that when one to one approach is adapted, no of keys generated are equal in no as that of files, i.e. one key for one file, which is not feasible for one to handle and share as well.
Whereas when key-agg. Encryption is adapted, no of keys generated/deployed is only one due to aggregation of keys performed. Here it is seen that no of keys are decreased to a great extent due to key aggregate encryption.

V. CONCLUSION

Users' knowledge (content/data) privacy indeed a central question of cloud storage. In this, we emphasize Compression of the secret keys in cryptosystems that support various cipher text categories in cloud storage. It could be any one of the ability set of categories, no issues; the delegatee will forever get combination key of constant size. In cloud storage, the quantity of cipher texts sometimes grows apace with none restrictions. Therefore we've to order enough cipher text categories for the longer term extension. Otherwise, we should expand the owner-key.

ACKNOWLEDGMENT

The authors herein thank the publishers, researchers for making their resources available, as well as professors for their advice and proper guidance. We also thank the college/institute authority for providing us the desired infrastructure and their honest support.

REFERENCES

[1] Cheng-Kang Chu, Sherman S. M, "Key Aggregate Cryptosystem for Scalable Data Sharing in cloud storage", IEEE Transactions on Parallel and Distributed Systems, vol. 25, issue2, 2014.
[2] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records," in Proceedings of ACM Workshop on Cloud Computing Security (CCSW '09). ACM, 2009, pp. 103–114.
[3] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted data," in Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06). ACM, 2006, pp. 89–98.
[4] F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-Identity Single-Key Decryption without Random Oracles," in Proceedings of Information Security and Cryptology (Inscrypt '07), ser. LNCS, vol. 4990. Springer, 2007, pp. 384–398.
[5] M. Chase and S.S.M. Chow, "Improving Privacy and Security in Multi-Authority Attribute-Based Encryption," Proceeding ACM Conference on Computer and Communication Security, pp. 121-130. 2009.
[6] S. G. Akl and P. D. Taylor, "Cryptographic Solution to a Problemof Access Control in a Hierarchy," ACM Transactions on Computer Systems (TOCS), vol. 1, no. 3, pp. 239–248, 1983.
[7] G. C. Chick and S. E. Tavares, "Flexible Access Control with Master Keys," in Proceedings of Advances in Cryptology – CRYPTO '89, ser. LNCS, vol. 435. Springer, 1989, pp. 316–322.
[8] 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.
[9] R. S. Sandhu, "Cryptographic Implementation of a Tree Hierarchy for Access Control," Information Processing Letters, vol. 27, no. 2, pp. 95– 98, 1988.
[10] Y. Sun and K. J. R. Liu, "Scalable Hierarchical Access Control in Secure Group Communications," in Proceedings of the 23th IEEE International Conference on Computer Communications (INFOCOM '04). IEEE, 2004.
[11] D. Boneh and M. K. Franklin, "Identity-Based Encryption from the Weil Pairing," in Proceedings of Advances in Cryptology – CRYPTO '01, ser. LNCS, vol. 2139. Springer, 2001, pp. 213–229.
[12] A. Sahai and B. Waters, "Fuzzy Identity-Based Encryption," in Proceedings of Advances in Cryptology - EUROCRYPT '05, ser. LNCS, vol. 3494. Springer, 2005, pp. 457–473.
[13] 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.