Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 6, Issue. 1, January 2017, pg.128 – 134

Key-Aggregate Searchable Encryption for Group Data Sharing via Cloud Storage

Ms. Chetna Waykole, Mr. D.D.Patil

Computer Department (SSGBCOET, Bhusawal), North Maharashtra University, Jalgaon, India
chetna.waykole@gmail.com, dineshonly@gmail.com

Abstract— Keyword search under the multi-tenancy setting is a more common scenario. In such a scenario, the data owner would like to share a document with a group of authorized users, and each user who has the access right can provide a trapdoor to perform the keyword search over the shared document, namely, the "multi-user searchable encryption" (MUSE) scenario. Some recent work focus to such a MUSE scenario, although they all adopt single-key combined with access control to achieve the goal. In, MUSE schemes are constructed by sharing the documents searchable encryption key with all users who can access it, and broadcast encryption is used to achieve coarse-grained access control. attribute based encryption (ABE) is applied to achieve fine-grained access control aware keyword search. As a result, in MUSE, the main problem is how to control which users can access which documents, whereas how to reduce the number of shared keys and trapdoors is not considered. Key aggregate searchable encryption can provide the solution for the latter, and it can make MUSE more efficient and practical. The aim of proposed work is to design an Extended Cryptographic Mechanism for Secured Data Sharing in Cloud using Multiple Keys. Data sharing is an important functionality in cloud storage. In this project, we show how to securely, efficiently, and flexibly share data with others in cloud storage. We describe new public-key crypto-systems that produce cipher texts such that efficient delegation of decryption rights for any set of cipher texts is possible.

Keywords— Searchable encryption, data sharing, cloud storage, data privacy

I. INTRODUCTION

Cloud storage is gaining popularity recently. In enterprise settings, we see the rise in demand for data outsourcing, which assists in the strategic management of corporate data. It is also used as a core technology behind many online services for personal applications. Nowadays, it is easy to apply for free accounts for email, photo album, file sharing and/or remote access, with storage size more than 25GB (or a few dollars for more than 1TB). Together with the current wireless technology, users can access almost all of their files and emails by a mobile phone in any corner of the world. Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication, which means any unexpected privilege escalation will expose all data. In a shared-tenancy cloud computing environment, things become even worse. Data from different clients can be hosted on separate virtual machines (VMs) but reside on a single physical machine. Data in a target VM could be stolen by instantiating another VM co-resident with the target one. Regarding availability of files,

there are a series of cryptographic schemes which go as far as allowing a third-party auditor to check the availability of files on behalf of the data owner without leaking anything about the data, or without compromising the data owners anonymity. Likewise, cloud users probably will not hold the strong belief that the cloud server is doing a good job in terms of confidentiality. A cryptographic solution, with proven security re-lied on number-theoretic assumptions is more desirable, whenever the user is not perfectly happy with trusting the security of the VM or the honesty of the technical staff. These users are motivated to encrypt their data with their own keys before uploading them to the server. Data sharing is an important functionality in cloud storage. For example, bloggers can let their friends view a subset of their private pictures; an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively share encrypted data. Of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server.

There is a rich literature on searchable encryption, including SSE schemes and PKES schemes. In contrast to those existing work, in the context of cloud storage, keyword search under the multi-tenancy setting is a more common scenario. In such a scenario, the data owner would like to share a document with a group of authorized users, and each user who has the access right can provide a trapdoor to perform the keyword search over the shared document, namely, the "multi-user searchable encryption" (MUSE) scenario. Some recent work focus to such a MUSE scenario, although they all adopt single-key combined with access control to achieve the goal. In, MUSE schemes are constructed by sharing the documents searchable encryption key with all users who can access it, and broadcast encryption is used to achieve coarse-grained access control. attribute based encryption (ABE) is applied to achieve fine-grained access control aware keyword search. As a result, in MUSE, the main problem is how to control which users can access which documents, whereas how to reduce the number of shared keys and trapdoors is not considered. Key aggregate searchable encryption can provide the solution for the latter, and it can make MUSE more efficient and practical.

II. LITERATURE SURVEY

A. Identity Bases Encryption (IBE):

IBE is a type of a public-key encryption. Identity string is set for encryption which is nothing but users public key. In IBE, master secret keys are generated by the private key generator and here the secret key is provided based on users identity. Sender wants to share files. So sender will encrypt the files by making use of user identity and public parameter and sends the files. Receiver will decrypt these files by making use of his secret key. But out of key-aggregation and IBE, only one assumes random oracles. Key aggregation is inhibited as keys to be aggregated will come from various identity. The disadvantages of this system is ciphertext size is non-constant and cost of storing ciphertext and transmitting it expensive.

B. Symmetric Key Encryption:

Benaloh proposed an encryption scheme, where a huge number of keys can be sent rapidly in a broadcast scenario. The key origin is as follows. Initially choose two prime numbers p and q for a composite module. At random, master secret key will be chosen. Dissimilar prime numbers will be allied with each class. A public system parameter is considered for which all the prime numbers will be put. The outcome of this is a constant size key. This method is designed for symmetric-key setting. So here the sender should encrypt files with corresponding secret keys which will not be feasible. The disadvantages of this system are both encryption and decryption is done by same key and encryptor should get corresponding key to encrypt files.

C. Attribute Based Encryption (ABE):

In Attribute Based Encryption method an attribute will be linked with cipher text. From master secret key, the secret key will be derived. This secret key is used to decrypt the files merely if all its connected attributes go after the rules. Before Attribute Based Encryption method was introduced, the user who wanted secret key must go to third party and proving he is real by providing his identity and then he was capable to decrypt the file. Later in ABE scheme the secret key of user was not allowed to a single centre. Instead it was authorized by independent authorities. But still this scheme has drawback i.e. no solidity on secret key. Here in this scheme there is linear increase in key size, with the increase in attributes. Disadvantages are (a) Decryption key size is non-constant. (b) Requires more space to store keys. (c) Decryption key size increases linearly. (d) Managing keys is expensive.

D. Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data:

Attribute-based encryption (ABE)allows each ciphertext to be associated with an attribute, and the master-secret key holder can extract a secret key for a policy of these attributes so that a ciphertext can be decrypted by this key if its associated attribute conforms to the policy. For example, with the secret key for the policy (2v3v6v8), one can decrypt ciphertext tagged with class 2, 3, 6, or 8. However, the major concern in ABE is collusion resistance but not the compactness of secret keys. Indeed, the size of the key often increases linearly with the number of attributes it encompasses, or the ciphertext-size is not constant.

E. Chosen-Ciphertext Secure Proxy Re-Encryption:

To delegate the decryption power of some ciphertexts without sending the secret key to the delegatee, a useful primitive is proxy re-encryption (PRE). A PRE scheme allows sender to delegate to the server(proxy) the ability to convert the ciphertexts encrypted under her public-key into ones for receiver. PRE is well known to have numerous applications including cryptographic file system. Nevertheless, sender has to trust the proxy that it only converts ciphertexts according to her instruction ,which is what we want to avoid at the first place. Even worse, if the proxy colludes with receiver, some form of senders secret key can be recovered which can decrypt senders(convertible) ciphertexts without receivers further help. That also means that the transformation key of proxy should be well protected. Using PRE just moves the secure key storage requirement from the delegatee to the proxy. It is, thus, undesirable to let the proxy reside in the storage server. That will also be inconvenient since every decryption requires separate interaction with the proxy. Using PRE just moves the secure key storage requirement from the delegatee to the proxy. It is thus, undesirable to let the proxy reside in the storage server. That will also be inconvenient since every decryption requires separate interaction with the proxy.

F. Dynamic and Efficient Key Management for Access Hierarchies:

We start by discussing the most relevant study in the literature of cryptography/security. Cryptographic key assignment schemes aim to minimize the expensive storing and managing secret keys for general cryptographic use. Utilizing a tree structure, a key Using KAC for data sharing in cloud storage. We call this as master-secret key to avoid confusion with the delegated key we will explain later. For simplicity, we omit the inclusion of a decryption algorithm for the original data owner using the mastersecret key. In our specific constructions, we will show how the know ledge of the master-secret key allows a faster decryption than using Extract followed by Decrypt. For a given branch can be used to derive the keys of its. Just, granting the parent key implicitly grants all the keys of its descendant nodes. proposed a method to generate a tree hierarchy of symmetric-keys by using repeated evaluations of pseudo random function/block-cipher on a fixed secret. The concept can be generalized from a tree to a graph. More advanced cryptographic key assignment schemes support access policy that can be modelled by an acyclic graph or a cyclic graph . Most of these schemes produce keys for symmetric-key cryptosystems, even though the key derivations may require modular arithmetic as used in public-key cryptosystems, which are generally more expensive than symmetric-key operations such as pseudorandom function. We take the tree structure as an example. Alice can first classify the ciphertext classes according to their subjects like Each node in the tree represents a secret key, while the leaf nodes represents the keys for individual ciphertext classes. Filled circles repre-sent the keys for the classes to be delegated and circles circumvented by dotted lines represent the keys to be granted. Note that every key of the non leaf node can derive the keys of its descendant nodes. if Alice wants to share all the files in the personal category, she only needs to grant the key for the node personal, which automatically grants the delegatee the keys of all the descendant nodes (photo, music). This is the ideal case, where most classes to be shared belong to the same branch and thus a parent key of them is sufficient.

G. Fuzzy Identity-Based Encryption. Theory and Applications of Cryptographic Techniques:

IBE 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 (e.g., an email address). There is a trusted party called private key generator in IBE which holds a master-secret key and issues a secret key to each user with respect to the user identity. The encryptor can take the public parameter and a user identity to encrypt a message. The recipient can decrypt this ciphertext by his secret key. Guest tried to build IBE with key aggregation. One of their schemes assumes random oracles but another does not. In their schemes, key aggregation is constrained in the sense that all keys to be aggregated must come from different identity divisions. While there are an exponential number of identities and thus secret keys, only a polynomial number of them can b e aggregated. Most importantly, their key aggregation comes at the expense of On sizes for both ciphertext and the public parameter, where is the number of secret keys which can be aggregated into a constant size one. This greatly increases the costs of storing and transmitting ciphertext, which is impractical in many situations such as shared cloud storage. As we 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 ciphertext encrypted under many identities which are close in a certain metric space.

There exist several expressive ABE schemes where the decryption algorithm only requires a constant number of pairing computations. Recently, Green et al. Proposed a remedy to this problem by introducing the notion of ABE with outsourced decryption, which largely eliminates the decryption overhead for users. Based on the existing ABE schemes Green et al. Also presented concrete ABE scheme with outsourced decryption. In this existing system, a user provides an untrusted server, say a proxy operated by a cloud service provider with a transformation key TK that allows the latter to transfer any ABE cipher text CT satisfied by that users attributes or access policies into a simple cipher text CT and it only incurs a small overhead for the user to recover the plain text from transformed cipher text CT. The security property of the ABE scheme with outsourced decryption guarantees that an adversary (including the malicious server) be not able to learn anything about the

encrypted message; however, the scheme provides no guarantee on the correctness of the transformation done by the cloud server. In the cloud computing setting, cloud service providers may have strong financial incentives to return incorrect answers, if such answers requires less work and are unlikely detected by users.

Drawbacks of Existing System are Symmetric Key Encryption, Key-Policy Attribute based Encryption, Relation between class required, Single cloud is merged, No guarantee on staff.

III. PROPOSED APPROACH

The proposed system is designed with an e client public-key encryption. In this any number of subset of the ciphertext can be decrypted by the decryption key. The problem is solved by the introduction of key aggregate cryptosystem. In key aggregate cryptosystem user will encrypt message not merely in a public key but also beneath an identifier. These ciphertexts are more characterized into classes. The owner will have the master secret key. The secret keys are extracted from the master secret key, these secrets keys are used to encrypt the les. The extracted key can be aggregate key which is as compact as a single key. By this solution, sender shares the single aggregate key by means of a safe channel say email. The receiver downloads the encrypted les from senders drop box and then decrypts those les with single aggregate key. This scenario is shown in Figure below:

KAC in meant for the data sharing. The data owner can share the data in desired amount with confidentiality. KCA is easy and secure way to transfer the delegation authority. For sharing selected data on the server Alice first performs the Setup. Later the public/master key pair (pk, mk) is generated by executing the KeyGen. The msk master key is kept secret and the public

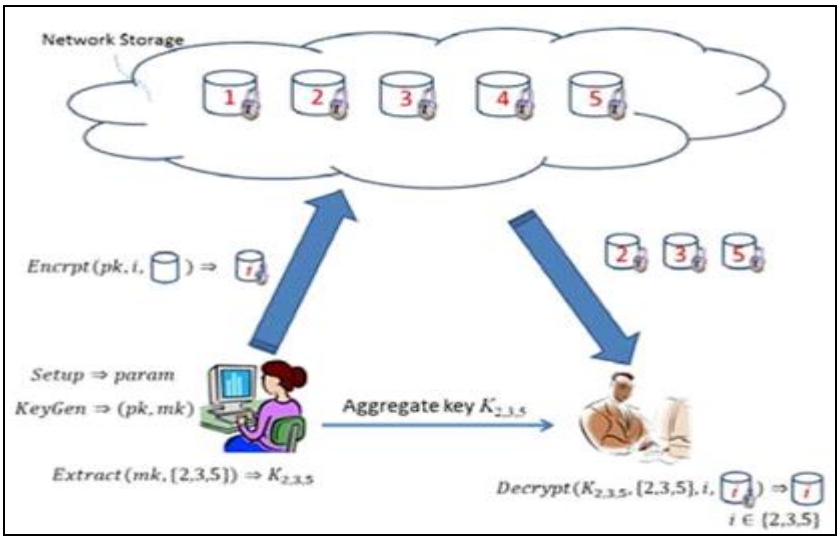

Fig.1 System Architecture

key pk and param are made public. Anyone can encrypt the data m and this data is uploaded on server. With the decrypting authority the other users can access those data. If Alice is wants to share a set S of her data with a friend Bob then she can perform the aggregate key KS for Bob by executing Extract (mk, S). As kS is a constant size key and the key can be shared through secure e-mail. When the aggregate key has got Bob can download the data and access it.

A. Algorithm:

1. Setup (1, n) The data owner establishes public system parameter via Setup. On input of a security level parameter 1 and number of cipher text classs n, it outputs the public system parameter param
2. KeyGen It is for generation of public or master key secret pair.
3. Encrypt (pk, i, m) It is executed by data owner and for message m and index i, it computes the cipher text as C.
4. Extract (msk, S) It is executed by data owner for delegating the decrypting power for a certain set of ciphertext classes and it outputs the aggregate key for set S denoted by Ks.
5. Decrypt (Ks,S,i,C)) When an appointee receives aggregate key Ks as exhibited by the previous step, it can execute Decrypt. The decrypted original message m is displayed on entering Ks, S, i, and C, if and only if I belongs to the set S

B. Aggregate Key Generation Algorithm

1. First Setup Data
2. All the key like k1,k2 ,k3 are in string format then it will converted into bytes using Byte Encoder.
3. Then every string converted in string to number like, K1=12356, K2=56423, K3=35641

4. All set key combine then it can give separator for that different key like,12356 0 56423 0 35641 here no value consider as separator.
5. secrete key i.e, S.
6. key convolution : we are use the quadratic equation, f(x)=(n1x + n2x + S)/n1=94,n2/66 here the x is consider as 2 or any number.
7. Then it calculation getting the number like 254631 then that no again converted in String.
8. Display String format of key.

C. Implementation Details:

AES Encryption Algorithm:

- After Files will be uploaded by a registered user will encrypted by using Improved AES Encryption technique then file will be stored on cloud.
- First we process file using base64 encoder to convert it in byte array.
- Then this byte array will be encrypted using AES technique.

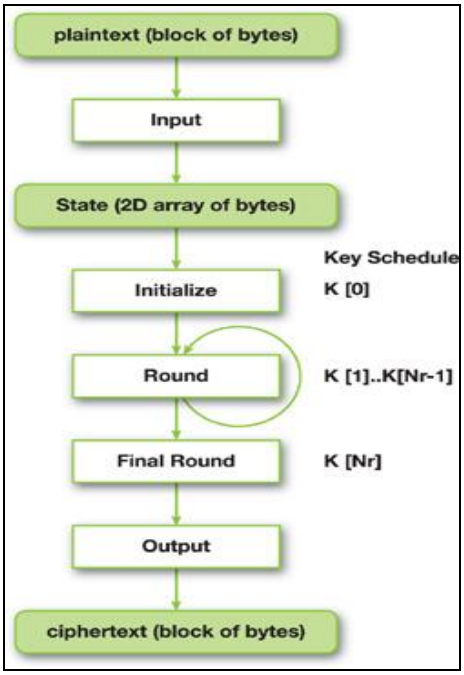

Fig. 2 AES Encryption Algorithm

IV. RESULTS

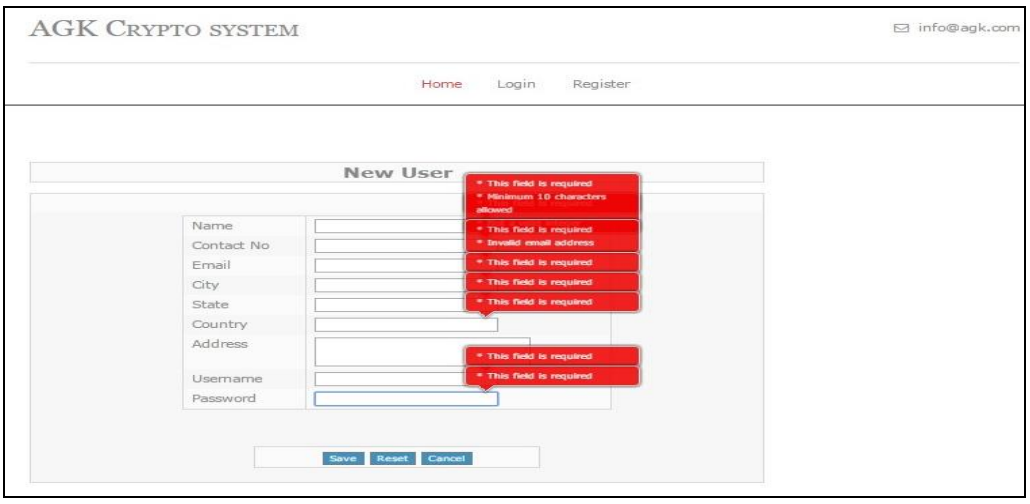

Fig. 3 User Registration

Fig. 4 Login Screen

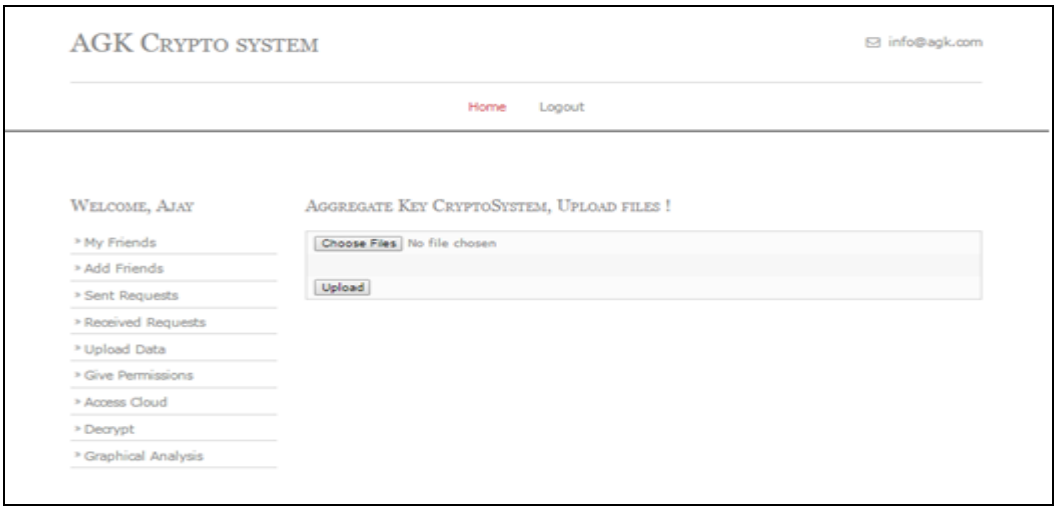

Fig. 5 Upload file

V. CONCLUSIONS

In this work we have reviewed three authentication techniques: Attribute based encryption (ABE), Identity Based Encryption (IBE) and Key Aggregate Cryptosystem (KAC). The major concern in ABE is collusion resistance but not compression of secret keys. Definitely, the ciphertext size is not constant. In IBE, random set of identities are not match with our design of key aggregation. Key Aggregate Cryptosystem protects users data privacy by compressing the secret key in public key cryptosystem which supports delegation of secret key for different cipher text classes. For future extension it is necessary to reserve enough cipher texts classes because in cloud cipher texts grows rapidly and the limitation is that bound of the number of maximum cipher text classes.

To share data exibly is vital thing in cloud computing. Users prefer to upload there data on cloud and among di erent users. Outsourcing of data to server may lead to leak the private data of user to everyone. Encryption is a one solution which provides to share selected data with desired candidate. Sharing of decryption keys in secure way plays important role. Public-key cryptosystems provides delegation of secret keys for different ciphertext classes in cloud storage

REFERENCES

[1] F. C. Chang and H. C. Huang, \Key-Aggregate Cryptosystem for Scalable Data Shar-ing in Cloud Storage," Inf. Sci., vol. 192, no. 1, pp. 3949, Jun. 2012.

[2] S. S. M. Chow, Y. J. He, L. C. K. Hui, and S.-M. Yiu, \SPICE - Simple Privacy-Preserving Identity-Management for Cloud Environment," in Applied Cryptography and Network Security .,ACNS 2012, ser. LNCS, vol. 7341. Springer, 2012, pp. 526543. pp. 173184, 2011.

[3] L. Hardesty, Secure computers arent so secure, \MIT press,"2009,http://www.physorg.com/news176107396.html.

[4] V. Goyal, O. Pandey, A. Sahai, and B. Waters, \Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data,Proc ," 13th ACM Conf. Computer and Comm. Security (CCS 06),pp. 89-98, 2006.
[5] S.S.M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, \Practical Leakage- Resilient Identity-Based Encryption from Simple Assumptions ," in Proc. ACM Conf. Com-puter and Comm. Security,pp. 152-161, 2010.
[6] G. Ateniese, A.D. Santis, A.L. Ferrara, and B. Masucci, \Provably-Secure Time-Bound Hierarchical Key Assignment Schemes, ,"J. Cryptology.,vol. 25, no. 2, pp. 243-270, 2012.
[7] F. Guo, Y. Mu, Z. Chen, and L. Xug, \Multi-Identity Single-Key Decryption without Random Oraclesl,",in Proceedings of Information Security and Cryptology (Inscrypt 07), ser. LNCS, vol. 4990. Springer, 2007, pp. 384398.
[8] Vigneshwaran.K 1, Sumithra.S2, Janani.R3 "An Intelligent Tracking System Based on GSM and GPS Using Smartphones" Vol. 4, Issue 5, May 2015.

