A Survey on Attribute-Based Encryption Technique for Scalable Data Sharing in Cloud Storage

[1]Mr.V.Janarthanan, Assistant Professor,
Department of MCA,
Vivekanandha College of Arts and Sciences for Women(Autonomous), Tiruchengode, Tamilnadu, India.

[2]Dr.R.Nandhakumar, Assistant Professor,
PG & Research Department of Computer Science & Applications,
Vivekanandha College of Arts and Sciences for Women(Autonomous), Tiruchengode, Tamilnadu, India.

***ABSTRACT*: Cloud computing provides the flexible architecture to share the applications as well as the other network resources. Cloud storage enables networked online storage. Key management and key sharing plays the main role in the data sharing concept of cloud computing. While cloud computing brings new and challenging security threats to the outsourced data. Hence data owner does not have any control to ensure data security in host data centers. In this paper, Cipher text policy attribute-based encryption (CP-ABE) is introduced with Key aggregate cryptosystem a promising cryptographic solution to this issue. It enables data owners to define their own access policies over user attributes and enforce the policies on the data to be distributed and Attribute policies, Cipher text policies are being aggregated. This effectively eliminates the need to rely on the data storage server for preventing unauthorized data access and integrity. Adding User revocation model enables removing access to suspicious users and Intrusion detection system to alert the owner during defined attacks noticed.**

Keywords: Key aggregate cryptosystem, access control, user revocation, intrusion detection.

I. INTRODUCTION

Cloud computing is the new trending model used for computing in which the internet is used for communicating and storing the data. Some of the most crucial functionalities of cloud computing is data sharing and securely storing the important data dumped into cloud. When it comes to sharing and storing of data, the users of the cloud become bit hesitant to put the data onto the cloud scaring about the confidentiality and security of the data. Due to these aspects of preserving the security and confidentiality of the data, the notion of encryption came into picture. Here the users can encrypt their data using various encryption algorithms before help of the third Party key generators for encrypting and decrypting of data or can encrypt by themselves using various algorithms.

Cloud storage is day- by-day gaining popularity. It is being utilized as core technology for various online services. The wireless technology enables use to access almost all files, emails and data for the users using their smart devices from any remote corner of the world. Data sharing is a prime functionality in the cloud storage. The blog writers usually allow their friends to have a look or access some of the confidential files among the various files dumped in the cloud; any organization may grant their employees to access a small part of their confidential data. So here the sharing of the encrypted data with only the authentic users, who are given the rights to access it, is the challenging factor. Although users have the option of downloading the encrypted data from the cloud, decipher them, and later send them to their friends for sharing it, but this will simply lessen the impact of cloud storage. Instead the authentic users must be given the privilege of rights for accessing while data sharing with others in such a way for accessing those data directly from the server.

Cloud Storage is a service where data is remotely maintained, managed, and backed up. This service is available to users over a network, which is usually the internet. It allows the user to store files online so that the user can access them from any location via the internet. While considering data privacy, we cannot rely on traditional technique of authentication, because unexpected privilege escalation will expose all data. Solution is to encrypt data before uploading to the server with user's own key.

Data sharing is again important functionality of cloud storage, because user can share data from anywhere and anytime to anyone. For example, organization may grant permission to access part of sensitive data to their employees. But challenging task is that how to share encrypted data. Traditional way is user can download the encrypted data from storage, decrypt that data and send it to share with others, but it loses the importance of cloud storage. Cryptography technique can be applied in a two major ways- one is symmetric key encryption and other is asymmetric key encryption. In symmetric key encryption, same keys are used for encryption and decryption. By contrast, in asymmetric key encryption different keys are used, public key for encryption and private key for decryption, also known as Public-key encryption.

Use of Public-key encryption is a powerful mechanism for protecting the confidentiality of stored and transmitted information is more flexible for our approach. Since the decryption key should be sent via secure channel and kept secret small size is always enviable.

Need for Key Aggregation

Key aggregation plays a prime role in overcoming the network overheads. On considering a scenario, where a particular user Alice wants to send an access key to her friend Bob, who wants to access some of the files. Alice has encrypted those files before uploading them onto the cloud. Then Alice can send an aggregate key of these corresponding secret keys of the various files using which Bob can decrypt them. Here, the load on network traffic is lowered, as the problem of sending all the corresponding keys is replaced by sending just a single aggregate key. The expenses of having a tamper proof storage are usually high. The cost of secured storage for storing these secret keys is also reduced by storing the aggregate key due to its compact size. To protect a user's data confidentiality, some form of access control needs to be implemented in the Cloud. Access control should allow a user to choose who can view his data and who shouldn't. Access Control Lists (ACLs) were originally used [10], however, it was not effective as it was too coarse-grained and was not scalable; one of the primary features of the Cloud.

Protection from Privacy and Security Attacks:

We propose the definition of direct revocation system which enables the owner to directly revoke any users at any time, and formulates the security model to distinguish dishonest cloud servers and unauthorized users through User revocation model. Motives of a malicious user-to steal valuable data, to cause controversy, to get revenge, to prove intellect and to gain prestige/ are just curious. To overcome this we use Intrusion detection system which lets the owner to know about malicious user attack. It alerts the owner by sending message about the attack noticed.

II. LITERATURE SURVEY

SYMMETRIC-KEY ENCRYPTION WITH COMPACT KEY

Benaloh et al. [2] presented an encryption scheme which is originally proposed for concisely transmitting large number of keys in broadcast scenario [3]. The construction is simple and we briefly review its key derivation process here for a concrete description of what are the desirable properties we want to achieve. The derivation of the key for a set of classes (which is a subset of all possible cipher text classes) is as follows. A composite modulus is chosen where p and q are two large random primes. A master secret key is chosen at random. Each class is associated with a distinct prime. All these prime numbers can be put in the public system parameter. A constant-size key for set can be generated. For those who have been delegated the access rights for S' can be generated. However, it is designed for the symmetric-key setting instead. The content provider needs to get the corresponding secret keys to encrypt data, which is not suitable for many applications. Because method is used to generate a secret value rather than a pair of public/secret keys, it is unclear how to apply this idea for public-key encryption scheme. Finally, we note that there are schemes which try to reduce the key size for achieving authentication in symmetric-key encryption, e.g., [4]. However, sharing of decryption power is not a concern in these schemes.

IBE WITH COMPACT KEY Identity-based encryption (IBE) (e.g., [5], [6], [7]) is a 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, mobile number). There is a private key generator (PKG) in IBE which holds a master-secret key and issues a secret key to each user with respect to the user identity.

The content provider can take the public parameter and a user identity to encrypt a message. The recipient can decrypt this cipher text by his secret key. Guo et al. [8], [9] tried to build IBE with key aggregation. 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 be aggregated.[1] This significantly increases the costs of storing and transmitting cipher texts, which is impractical in many situations such as shared cloud storage. As 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.[1] we mentioned, our schemes feature constant cipher text size, and their security holds in the standard model. In fuzzy IBE [10], one single compact secret key can decrypt cipher texts 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.

TABLE I
COMPARISONS BETWEEN THE BASIC KAC SCHEME AND OTHER RELATED SCHEMES

Different Schemes	Cipher text size	Decryption key size	Encryption type
Key assignment schemes	Constant	Non-constant	Symmetric or public-key
Symmetric- key encryption with compact key	Constant	Constant	Symmetric key
IBE with compact key	constant	Constant	Public key
Attribute based encryption	Constant	Non-constant	Public key
KAC	Constant	Constant	Public key

ATTRIBUTE-BASED ENCRYPTION

Attribute- based encryption (ABE) [11], [12] allows each cipher text 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 cipher text can be decrypted by this key if its associated attribute conforms to the policy. For example, with the secret key for the policy (1 ˈ 3 ˈ 6 ˈ 8), one can decrypt cipher text tagged with class 1, 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 cipher text-size is not constant.

The importance of data sharing and the need to ensure privacy and security is discussed in a number of existing articles. Saradhy and Muralidhar [11] review the impact of the Internet on data sharing across many different organizations such as government agencies and businesses. They classify data sharing into data dissemination, query restriction, and record matching. They also provide a framework for secure and useful sharing of data on the internet. Butler [12] describes the issues of data sharing on the Internet where sharing information can allow users to infer details about users. This is useful as it raises awareness to organizations that the data they choose to share with the public can still raise privacy issues and does not guarantee the confidentiality of its users. We provide an Intrusion detection system which alerts the owner on attacks. And User revocation model to periodically audit the user actions, to revoke any malicious users from cloud storage access.

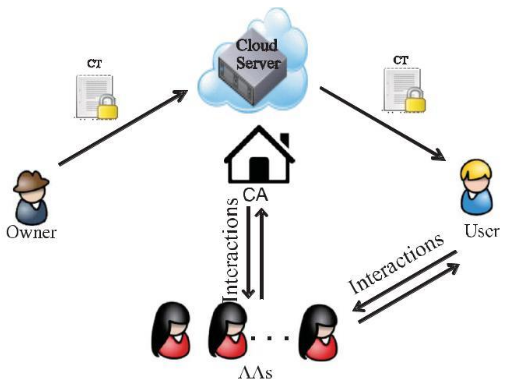

Fig. 1.System model.

III. SYSTEM MODEL AND SECURITY ASSUMPTIONS

In this section, we give the definitions of the system model, the security assumptions and requirements of our public cloud storage access control.

A. System Model

The system model of our design is shown in Fig. 1, which involves five entities: a central authority (CA), multiple attribute authorities (AAs), many data owners (Owners), many data consumers (Users), and a cloud service provider with multiple cloud servers(here, we mention it as cloud server.).

- The central authority (CA) is the administrator of the entire system. It is responsible for the system construction by setting up the system parameters and generating public key for each attribute of the universal attribute set. In the system initialization phase, it assigns each user a unique U i d and each attribute authority a unique Ai d. For a key request from a user, CA is responsible for generating secret keys for the user on the basis of the received intermediate key associated with the user's legitimate attributes verified by an AA. As an administrator of the entire system, CA has the capacity to trace which AA has incorrectly or maliciously verified a user and has granted illegitimate attribute sets
- The attribute authorities (AAs) are responsible for performing user legitimacy verification and generating intermediate keys for legitimacy verified users. Unlike most of the existing multi-authority schemes where each AA manages a disjoint attribute set respectively, our proposed scheme involves multiple authorities to share the responsibility of user legitimacy verification and each AA can perform this process for any user independently. When an AA is selected, it will verify the users' legitimate attributes by manual labor or authentication protocols, and generate an intermediate key associated with the attributes that it has legitimacy-verified. Intermediate key is a new concept to assist CA to generate keys.
- The data owner (Owner) defines the access policy about who can get access to each file, and encrypts the file under the defined policy. First of all, each owner encrypts his/her data with a symmetric encryption algorithm. Then, the owner formulates access policy over an attribute set and encrypts the symmetric key under the policy according to public keys obtained from *CA*. After that, the owner sends the whole encrypted data and the encrypted symmetric key (denoted as cipher text *C T*) to the cloud server to be stored in the cloud.
- ***The data consumer (User)*** is assigned a global user identity *U i d* by *CA*. The user possesses a set of attributes and is equipped with a secret key associated with his/her attribute set. The user can freely get any interested encrypted data from the cloud server. However, the user can decrypt the encrypted data if and only if his/her attribute set satisfies the access policy embedded in the encrypted data.
- ***The cloud server*** provides a public platform for own-ers to store and share their encrypted data. The cloud server doesn't conduct data access control for owners. The encrypted data stored in the cloud server can be downloaded freely by any user.

B. Security Assumptions and Requirements

In our proposed scheme, the security assumptions of the five roles are given as follows. The cloud server is always online and managed by the cloud provider. Usually, the cloud server and its provider are assumed to be "honest-but-curious", which means that they will correctly execute the tasks assigned to them for profits, but they would try to find out as much secret information as possible based on data owners' inputs and uploaded files. *CA* is the administrator of the entire system, which is always online and can be assumed to be fully trusted. It will not collude with any entity to acquire data contents. *AA*s are responsible for conducting legitimacy verification of users and judging whether the users have the claimed attributes. We assume that *AA* can be compromised and cannot be fully trusted.

Furthermore, since the user legitimacy verification is conducted by manual labour, mis-operation caused by carelessness may also happen. Thus, we need an auditing mechanism to trace an *AA*'s misbehavior. Although a user can freely get any encrypted data from the cloud server, he/she cannot decrypt it unless the user has attributes satisfying the access policy embedded inside the data. Therefore, some users may be dishonest and curious, and may collude with each other to gain unauthorized access or try to collude with (or even compromise) any *AA* to obtain the access permission beyond their privileges.

Owners have access control over their uploaded data, which are protected by specific access policies they defined.

To guarantee secure access control in public cloud storage, we claim that an access control scheme needs to meet the following four basic security requirements:

- *Data confidentiality*. Data content must be kept confidential to unauthorized users as well as the curious cloud server.
  - *Collusion-resistance*. Malicious users colluding with each other would not be able to combine their attributes to decrypt a cipher text which each of them cannot decrypt alone.
- AA *accountability*. An auditing mechanism must be devised to ensure that an *AA*'s misbehavior can be detected to prevent *AA*s' abusing their power without being detected.
- *No ultra vires for any* AA. An *AA* should not have unauthorized power to directly generate secret keys for users. This security requirement is newly introduced based on our proposed hierarchical framework.

IV.METHEDOLOGY

The aggregation cryptosystem consists of efficient Key Aggregate Cryptosystem algorithm. The data owner set up the general public parameter using Setup and creates a public/private(master) key and combines using KeyGen.

The secret file is encrypted utilizing CP-ABE algorithm. The information owner will make use the master-secret to come up with aggregate decipher key for a collection of data files.

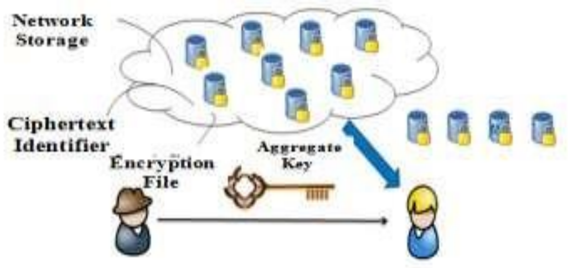

Figure. 1. Using KAC for data sharing in cloud storage.

The generated keys may be passed to delegates securely (via secure e-mails or secure devices). Finally, any user with aggregate key will decrypt data file and download it.

A canonical application of KAC is data sharing. The key aggregation property is especially useful when we expect delegation to be efficient and flexible. The KAC schemes enable a content provider to share data in a confidential and selective way, with a fixed and small cipher text expansion, by distributing to each authorized user a single and small aggregate key. We study how to make a decryption key more powerful in the sense that it allows decryption of multiple cipher texts, without increasing its size. In KAC, users encrypt a message not only under a public-key, but also under an identifier of cipher text called class. That means the cipher texts are further categorized into different classes. The key owner holds a secret called master-secret key, which can be used to extract secret keys for different classes. More importantly, the extracted key have can be an aggregate key which is as compact as a secret key for a single class,but aggregates the power of many such keys, i.e., the decryption power for any subset of cipher text classes.

Data sharing in cloud storage using KAC, illustrated in Figure 1. Suppose Alice wants to share her data $m1,m2,....,m_n$ on the server. She first performs Setup (1^{λ}, n) to get param and execute KeyGen to get the public/master-secret key pair (pk, msk). The system parameter param and public-key pk can be made public and master-secret key msk should be kept secret by Alice. Anyone can then encrypt each mi by Ci = Encrypt (pk, i, mi). The encrypted data are uploaded to the server. With param and pk, people who cooperate with Alice can update Alice's data on the server. Once Alice is willing to share a set S of her data with a friend Bob, she can compute the aggregate key KS for Bob by performing Extract (msk, S). Since KS is just a constant size key, it is easy to be sent to Bob through a secure e-mail. After obtaining the aggregate key, Bob can download the data he is authorized to access. That is, for each i ϵ S, Bob downloads Ci from the server. With the aggregate key KS, Bob can decrypt each Ci by Decrypt (K_S, S, i, Ci) for each i ϵ S.

V.ALGORITHM

Input: Data files.

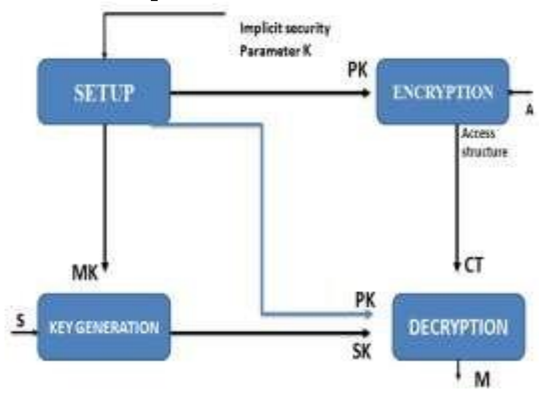

Fig. 2: Algorithm flowchart

Step 1: Setup (1^{λ}, n) - In this step the data owner encrypts the data files (pdf, txt,doc). On input a security level parameter 1^{λ} and the number of data file classes n (i.e., each file having private key should be generated by using public and private key pair), it outputs the public system parameter param, using Data encryption standard algorithms for security purpose.

Step 2: KeyGen - It is executed by data owner to randomly generate a public/ master-secret key pair (P_k, msk).

Step 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.

Step 4:Extract (msk, S) - It is executed by data owner for delegating the decrypting power for acertain set of cipher text classes and it outputs the aggregate key for set S denoted by SK.

Step 5:Decrypt (Ks, S, I, C) - It is executed by a delegate who received, an aggregate key Ks generated by Extract. On input Ks, set S, an index i denoting the cipher text class cipher text C belongs to and output is decrypted result m.

Output: Downloaded file

VI.CONCLUSION

In this paper, we discussed the public-key encryption methodology for protecting the privacy of data from the attackers who may obtain the data by legal or other means, data stored by users and confidential information. The main aim of this approach is to obtain the aggregate key of constant size empowered with the decryption rights for the number of files is possible by the valid user. Along with the privacy of data, the confidentiality is also preserved by encrypting the user data before dumping into the cloud.

Protection of the users' data privacy in cloud storage is an important aspect. With the help of mathematical tools, the encryption schemes are becoming more versatile and have started involving many encryption and decryption keys for a single application. But this project introduced the unique concept of the aggregation of the keys involved in decryption process. The cost of storing and transmitting the cipher texts is lowered as they are constant-sized. This compact aggregate key can be conveniently sent to others or be stored in a smart card with very limited secure storage. It is modeled in such a way keeping different security levels and extensions.

Storing the delegated keys in the mobile devices which have no trusted software, there is a possibility that the key gets disclosed. So designing a leakage-proof cryptosystem which supports flexible and efficient key delegation is an interesting direction. In this project, the CP- ABE algorithm is used for encrypting the files. A more secured and efficient algorithm can be used in future so as to cope up with the speed and for security purpose. So we have to reserve enough cipher text classes for the future extension. Otherwise, we need to expand the public-key. Although the parameter can be downloaded with cipher texts, it would be better if its size is independent of the maximum number of cipher text classes.

VII.REFERENCES

[1] Cheng-Kang Chu ,Chow, S.S.M, 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. Volume: 25, Issue: 2. Year :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] J. Benaloh, ―Key Compression and Its Application to Digital Fingerprinting,‖ Microsoft Research, Tech. Rep., 2009.

[4] B. Alomair and R. Poovendran, ―Information Theoretically Secure Encryption with Almost Free Authentication,‖ J. UCS, vol. 15, no. 15, pp. 2937–2956, 2009.

[5] 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.

[6] 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.

[7] 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.

[8] F. Guo, Y. Mu, and Z. Chen, ―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] 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.

[10] Li J, Zhao G, Chen X, Xie D, Rong C, Li W, Tang L, Tang Y (2010) Fine-grained data access control systems with user accountability in cloud computing. IEEE second international conference on cloud computing technology and science(CloudCom) 2010, pp 89–96.

[11] Sarathy R, Muralidhar K (2006) Secure and useful data sharing. Decis Support Syst 204–220.

[12] ButlerDData sharing threatens privacy, vol 449 (7163).Nature Publishing, Group, pp 644–645.

[13] Google search engine www.google.com – for conceptual search.

[14] Danan Thilakanathan, Shiping Chen, Surya Nepal and Rafael A. Calvo - Secure Data Sharing in the Cloud.

[15] LI Shuanbao123, FU Jianming121- user revocation for data sharing based on Broadcast CP-ABE in cloud computing.