# Encryption Schemes for Scalable Cloud Storage

[1] Prof. Anup S. Kunte, [2] Sukhada S. Jadhav
[1] Assistant Professor & Head Department IT, [2] PG Student
[1]Information Technology, [2] Computers
[1]HOD, KGCE, Mumbai, India, [2] Student of YTGOIFOE, Mumbai, India

***Abstract*— Cloud system is used to provide large number of storage servers, which provide long-term storage service over the Internet. Cloud is having its feature scalability which provides the scalable cloud storage; varying data storage size that means an owner can increase the data size of the storage for the particular user. Constructing centralized storage system for the cloud system makes hackers stole the data easily for which general encryption schemes protects the data confidentiality. In our study a secure distributed storage system is formulated by integrating an efficient public-key encryption scheme. It supports flexible delegation in the sense that any subset of the cipher texts which is produced by the encryption scheme and is decryptable by a constant-size decryption key generated by the owner of the master-secret key.**

***Index Terms*— Cloud computing, encrypted data sharing, Key aggregate cryptosystem, Scalable cloud storage.**

---

## I. INTRODUCTION

Cloud computing is known as an alternative to traditional Information Technology due to its essential resource-sharing and low-maintenance characteristics. In the cloud computing, the (CSPs) cloud service providers, such as Amazon, provides various services to cloud users with the help of powerful data centers. By transferring local data management systems into cloud servers system, users can enjoy high quality services and save significant investments on their local computing infrastructures.

Cloud is having the ability to service a theoretical number of users called its scalability. At cloud storage we will get a fixed size storage space while the scalable cloud storage have varying data storage size that means an owner can increase the data size of the storage for the particular user. With the help of this ability, cloud can scale down as well as up as per required demands.

An essential service provided by most cloud providers is data storage. For example, a company allows its staffs in the same department to store and share their files in the cloud. By utilizing the cloud, the staffs can be completely released from the troublesome local data storage and maintenance. But, it poses a major risk to the confidentiality of stored files. Cloud servers managed by cloud providers are not fully trusted by its users as the data files stored in the cloud might be sensitive and confidential. To preserve data privacy, a basic solution is encrypting data files and then uploading those encrypted data to the cloud.

Designing an efficient and secure data sharing scheme in the cloud is not an easy task due to the following challenging issues. Primary, the identity privacy is most significant obstacles for the large deployment of cloud computing. Without the guarantee of identity privacy, users may be unwilling to join in cloud computing systems because their real identities could be easily discovered by cloud providers and attackers. On the other hand, unconditional identity privacy may experience the abuse of privacy. In previous example, a misbehaved staff can deceive others by sharing false files without being traceable. Therefore, traceability, which enables the manager to reveal the real identity of a user, is also highly desirable.

To provide proper services to user, cloud has to achieve this goal and minimize threats and vulnerabilities during cloud operation. As well as cloud systems can be used to enable data sharing capabilities and this can provide number of benefits to the user. With multiple users from different organizations contributing to data in the Cloud, the time and cost will be much less compared to having to manually exchange data and therefore creating a clutter of redundant and possibly out-of-date documents.

With social networking services such as Facebook, the benefit of sharing data are numerous such as the ability to share photos, videos, information and events, creates a sense of enhanced enjoyment in one's life and can enrich the lives of some people as they are amazed at how many people are interested in their life and well-being. For students and group-related projects, there has been a major importance for group collaborative tools. Google Docs provides data sharing capabilities as groups of students or teams working on a project can share documents and can work together with each other effectively. This allows higher productivity compared to previous methods of continually sending updated versions of a document to members of the group via email attachments. Also in modern healthcare environments, healthcare providers are willing to store and share electronic medical records via cloud and hence remove the geographical dependence between healthcare provider and patient. The sharing of medical data allows the remote monitoring and diagnosis of patients without the patient having to leave their house.

## II. EXISTING SYSTEM

This section gives the scenario about an existing system where we take Dropbox as an example for illustration. Assuming that Alice keeps all her private photos on Dropbox, and she doesn't want to expose her photos to anyone. Due to various data leakage possibilities, Alice cannot feel relieved by just relying on the privacy protection mechanisms provided by Dropbox, so she encrypts all the photos using her own keys before uploading to Dropbox. One day, Alice's friend, Bob, asks her to share the photos in which Bob appeared. Then Alice can use the share function of Dropbox. But problem is how to give the decryption rights for only these photos to Bob. A possible option Alice can choose is to securely send the secret keys involved to Bob. Naturally, there are some extreme ways for her under the traditional encryption paradigm:

- Alice will encrypt her all files with a single encryption key and then gives Bob the corresponding secret key directly.

  This method is inadequate since all un-chosen data may be also leaked to Bob.

- Alice will encrypt her files with different keys and sends Bob the corresponding secret key.

The numbers of such keys are as many as the number of the shared photos, say, a thousand. Transferring these all secret keys inherently requires a secure channel, and storing these keys requires rather expensive secure storage. With the increase in the number of the decryption keys to be shared, the costs and complexities involved get increases.

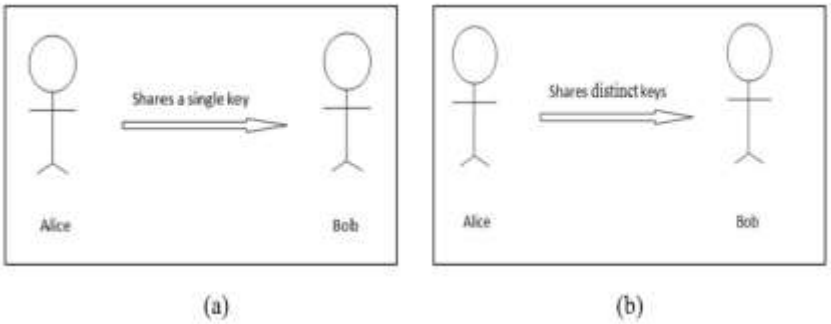


Fig 2.1 Dropbox scenarios (a) single key sharing, (b) distinct key sharing

Therefore, the best solution for the above problem is that Alice encrypts files with distinct public keys, but sends only a single constant-size decryption key to Bob. Since the decryption key should be sent via a secure channel and kept secret, small key size is always desirable for this. For instance, we cannot expect large storage for decryption keys in the resource-constraint devices like smart phones, smart cards etc.

These secret keys are usually stored in the tamper-proof memory, which is comparatively expensive. The present research efforts mainly focus on minimizing the communication requirements such as bandwidth, rounds of communication like an aggregate signature.

As per study the system will be used to design "An efficient public-key encryption scheme that supports flexible delegation in the sense that any subset of the cipher texts produced by the encryption scheme is decryptable by a constant-size decryption key generated by the owner of the master secret key (msk)".

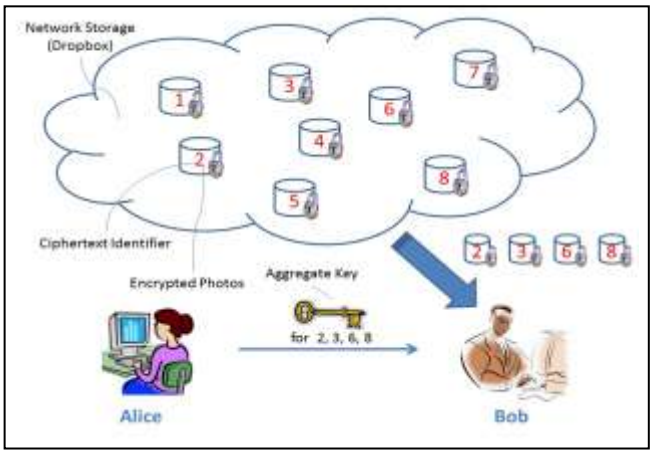


Fig 2.2 Alice shares files with identifiers 2, 3, 6 and 8 with Bob by sending him a single aggregate key [1]

This study introduced a special type of public-key encryption which we call Key-Aggregate Cryptosystem (KAC). In KAC, user encrypts a message not only under a public-key, but also under an identifier of cipher text i.e. class. That means the cipher texts are further categorized into different classes. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for different classes.

The extracted key have can be an aggregate key which is as compact as a secret key for a single class, but it aggregates the power of many such keys, i.e., the decryption power for any subset of cipher text classes. With the help of proposed solution, Alice will simply able to send Bob a single aggregate key through the secure e-mail. Then Bob will download the encrypted photos from Alice's Dropbox and use this aggregate key to decrypt these encrypted photos. The scenario is shown in Fig 2.2

The sizes of cipher text, public-key, master -secret key and aggregate key in our KAC schemes are all of constant size.

## III. LITERATURE SURVEY

There are several security schemes for data sharing on untrusted servers have been proposed. Few of are studied as follow.

Methodologies/Approaches

Below we describe different approaches for the processes of encryption in cloud computing environment.

### *A. Cryptographic Keys for a Predefined Hierarchy*

This scheme aims to minimize the expense in storing and managing secret keys for general cryptographic use. This utilizes a tree structure where a key for a given branch can be used to derive the keys of its descendant nodes. Thus granting the parent key implicitly grants all the keys of its descendant nodes. Sandhu [7] in his paper proposed a method to generate a tree hierarchy of symmetric keys by using repeated evaluations of pseudo random method or block-cipher on a fixed secret.

*Disadvantages over KAC*

1) Assigning key to different nodes to be shared causes to increase in the total key size.

2) Classifications become more complex when Alice wants to share different set of files to different people; thus this approach is not flexible.

3) Number of keys increases with increase in the number of branches.

For this delegatee in KAC scheme, the number of granted secret keys becomes the same as the number of classes.

### *B. Compact Key in Symmetric-Key Encryption*

This scheme is also motivated by the same problem of supporting flexible hierarchy in decryption power delegation but with symmetric-key setting. Benaloh et al. [2] has presented an encryption scheme which is originally proposed for transmitting large number of keys in broadcast scenario.

The construction is simple where a composite modulus $N = p \cdot q$ is chosen in which p and q are two large random prime numbers. Here a master secret key is chosen at random . Each class is associated with a distinct prime number . All these prime numbers can be put in the public system parameter and thus a constant-size key for set S′ can be generated.

*Disadvantages over KAC*

1) This approach achieves similar properties and performances as KAC scheme. But, it is designed for the symmetric-key setting. Thus an encryptor needs to get the corresponding secret keys to encrypt data, which is not suitable for many applications.

2) Since this 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.

3) Sharing of decryption power is not a concern in these schemes.

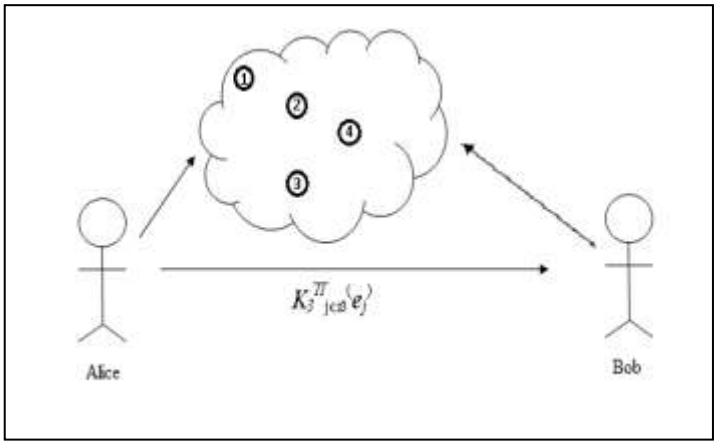


Fig.3.1. Alice sending a computed symmetric key to Bob

*C. Compact Key in Identity-Based Encryption*

Identity-based encryption (IBE) is a type of public-key encryption. In this scheme, 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 (PKG) in IBE which holds a master-secret key and issues a secret key to each user with respect to the user identity. The encryptor uses the public parameter and a user identity for encrypting his message. Then recipient decrypts this cipher text by his secret key.

In this scheme, key aggregation is constrained in the sense that all keys to be aggregated must come from different "identity divisions". Though there are an exponential number of identities and thus secret keys, only a polynomial number of them can be aggregated.

*Disadvantages over KAC*

1) The key-aggregation [9] comes at the expense of O(n)sizes for both cipher texts and the public parameter, where n is the number of secret keys which can be aggregated into a constant size one.

   This greatly increases the costs of storing and transmitting cipher texts, which is impractical in many situations such as shared cloud storage.

2) In fuzzy IBE [11], one single compact secret key can decrypt cipher a text 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 the idea of key aggregation.

KAC scheme feature constant cipher text size, and their security holds in the standard model.

*D. Attribute-based encryption*

Attribute-based encryption (ABE) [10] is an extended scheme of Identity-Based Encryption scheme. It is also a type of public-key encryption where the secret key of a user and the cipher text are dependent upon attributes.

In such a system, the decryption of a cipher text is possible only if the set of attributes of the user key matches the attributes of the cipher text.

It 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. Hence, 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 (2 ∨ 3 ∨ 6 ∨ 8), one can decrypt cipher text tagged with class 2, 3, 6 or 8.

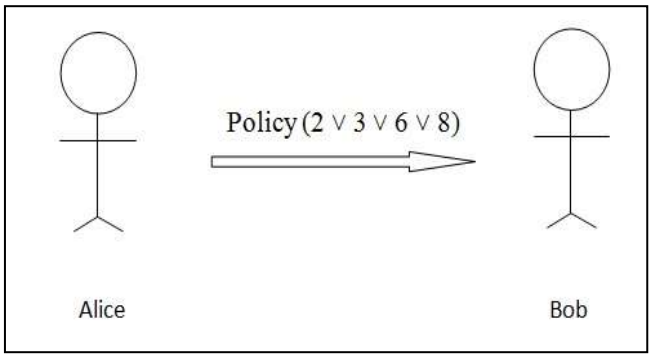


Fig.3.2.Attribute Based Encryption Scheme

*Disadvantages over KAC*

1) The major concern in ABE is collusion-resistance but not the compactness of secret keys.

2) Indeed, the size of the key often increases linearly with the number of attributes it encompasses, or the cipher text-size is not constant.

*E. Comparison*

The comparison of KAC scheme with other schemes which provides possible solutions on sharing in secure cloud storage is as follow.

Table 3.1
Comparison of various encryption schemes in secure cloud storage.

| Schemes | Decryption Key Size | Cipher Text Size | Encryption Type | Decryption Type | Data Security | Amount of data exchanged | Limitations |
|---|---|---|---|---|---|---|---|
| **Key Assignment Scheme[7]** | Depends on the hierarchy, therefore not constant | Constant | Symmetric or public key | Derived from descendent nodes. | Good | According to the number of keys for cipher text classes to be delegated. | Complex Classification, Not a flexible approach. |
| **Symmetric Key Encryption [2]** | Constant | Constant | Symmetric key | Distinct key according to subset | Good | As per the number of keys to be transmitted | Sharing of decryption power is not a concern |
| **Identity Based Encryption [11]** | Constant | Non-Constant | Public key | Identity based key generation | strong | Number of keys with respect to user identity and Public data parameters | Increases the costs of storing and transmitting cipher texts. |
| **Attribute Based Encryption [10]** | Non-Constant | Constant | Public key | Attribute based key generation | strong | Key for policy of attributes | The compactness of secret keys is not a concern. |
| **KAC** | Constant | Constant | Public key | Aggregate key | strong | Single aggregate key | A limitation in these is predefined bound of the number of maximum cipher text classes |

*Discussion:*

As mentioned in the Table 1, the Key assignment scheme is not flexible when the classifications gets more complex and one wants to share different sets of files to different people. For this delegatee in proposed scheme example, the number of granted secret keys becomes the same as the number of classes. In Symmetric key encryption achieves similar properties and performance

as proposed scheme but this scheme is designed for the symmetric key settings thus it cannot used to generate pair of public-secret keys and it is unclear that how to apply this scheme for public key encryption.

In Identity based Encryption (IBE) size of the key used for decrypting the data is constant but the size of encrypted data is not constant. As key is being generated with respected to particular user; these scheme provides better data security. While there are exponential number of identities and thus secret keys, only polynomial number of them can be used. This greatly increased the cost of transmitting cipher text. In Attribute based encryption scheme, the size of the key often increases linearly with the number of attributes it encompasses; therefore, the cipher text is not constant in this scheme.

Limitation of all above encryption scheme is overcome in proposed scheme KAC, a special type of public key encryption. In this scheme, the sizes of cipher text, public key, master-secret key and an aggregate key are constant. The proposed approach is flexible one because the constraint is eliminated that no special relation is required between the classes. One benefit of this scheme is that the decryption only takes two pairing while only one of them involves the aggregate key. It helps the fast computation.

## IV. RECOMMENDATION

### *A. K EY -AGGREGATE ENCRYPTION framework*

A key-aggregate encryption scheme consists of five polynomial-time algorithms as follows. The data owner establishes the public system parameter via Setup and generates a public/master-secret key pair via KeyGen. Messages can be encrypted via Encrypt by anyone who also decides what cipher text class is associated with the plaintext message to be encrypted.

The data owner can use the master-secret to generate an aggregate decryption key for a set of cipher text classes via Extract. The generated keys can be passed to delegatees securely (via secure e-mails or secure devices). Finally, any user with an aggregate key can decrypt any cipher text provided that the cipher text's class is contained in the aggregate key via Decrypt

*Setup (1 ^λ; n):*

Executed by the data owner to setup an account on an untrusted server.

Setup (1 ^λ; n) ⟶ param

Where,

1 ^λ = security level parameter
n = no. of cipher text classes
param = the public system parameter

*KeyGen:*

Executed by the data owner to randomly generate a public/master-secret key pair (pk; msk).

KeyGen ⟶ (pk,msk)

Where,
pk = public key
msk = master-secret key

*Encrypt (pk; i;m):*

Executed by anyone who wants to encrypt data.

Encrypt (pk; i;m) ⟶ $C^{'}$

Where,
pk = public key
i = index of cipher text class
m = message
$C^{'}$ = cipher text

*Extract (msk; S):*

Executed by the data owner for delegating the decrypting power for a certain set of cipher text classes to a delegatee

Extract (msk; S): ⟶ KS

Where, msk = master secret key
S = set indices of different classes
KS = the aggregate key for set S

*Decrypt (KS; S; i; C̃):*

Executed by a delegatee who received an aggregate key KS generated by Extract.

Decrypt (KS; S; i; C̃) ⟶ decrypted m

Where,
KS = the aggregate key for set S
S = set indices of different classes
i = index of cipher text class
C̃ = cipher text

*B. Sharing Encrypted Data*

A typical application of KAC is data sharing. The key aggregation property is especially useful when we expect the delegation to be efficient and flexible. The scheme enable a content provider to share the data in a confidential and selective way, with a fixed and small cipher text expansion, by distributing a single and small aggregate key to each authorized user. Below Figure 2 depict the main idea of data sharing in cloud storage using KAC.

Suppose Alice wants to share her data m1, m2,…,mv on the server. She first performs Setup (1 ^λ, n) to get param and then execute the 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 (including Alice herself)can then encrypt each mi by C̃i = Encrypt(pk, i, mi) which will be 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 via a secure e-mail or other channel. Once aggregate key is obtained, Bob can download the data to which he is authorized to access. That means, for each i∈S, Bob will download C̃i and some required values in param from the server. With the aggregate key KS, Bob can decrypt each C̃i by Decrypt (KS, S, i, C̃i) for each i ∈ S.

*C. Application*

Our study provides a candidate solution for the missing piece, public-key PCE for flexible hierarchy, which the existence of an efficient construction was an open question. Any patient can either define her own hierarchy according to her need, or follow the set of categories suggested by the electronic medical record system she is using, such as "clinic visits", "x-rays", "allergies", "medications" and so on. When the patient wishes to give access rights to her doctor, she can choose any subset of these categories and issue a single key, from which keys for all these categories can be computed.

Thus, we can essentially use any hierarchy we choose, which is especially useful when the hierarchy can be complex. Finally, one healthcare personnel deals with many patients and the patient record is possible stored in cloud storage due to its huge size (e.g., high resolution medical imaging employing x-ray), compact key size and easy key management are of paramount importance.

## V. CONCLUSION

We come to know that data privacy is a central question of any cloud storage system. With better mathematical tools, cryptographic schemes are getting more versatile. We use Key Aggregate Cryptosystem for secure and efficient data sharing in cloud storage system. This helps to make decryption key more powerful so that it can allow decryption of multiple cipher text without increasing its size.

By overcoming limitations of existing encryption schemes KAC enables a content provider to share user's data in a confidential and selective way along with fixed and small cipher text expansion by distributing a single aggregate key to each authorized user.

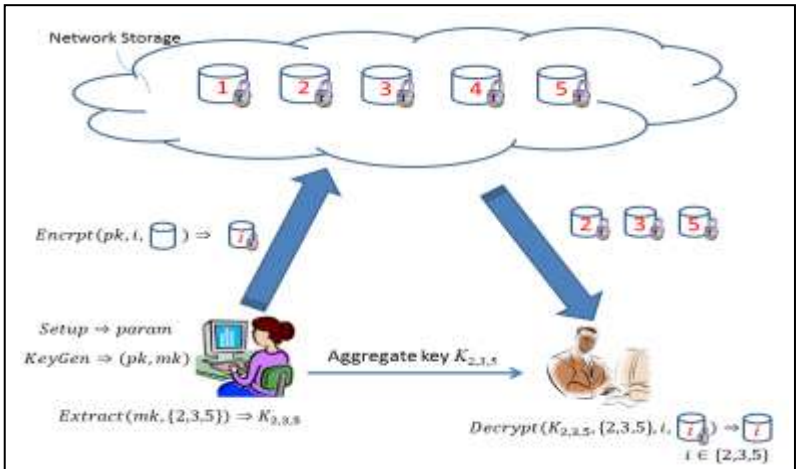


Fig.4.1.Using KAC for data sharing in cloud storage [1]

*FUTURE WORK*

This proposed scheme is having some limitations as the predefined bound of the number of maximum cipher text classes; because in cloud storage huge amount of data is being outsourced thus the number of cipher text also grows rapidly. Therefore we have to reserve enough cipher text classes for future expansion. If such situation takes place, we need to extend respected public keys.

Another drawback of this scheme is that if user carries his delegated keys over mobile devices without any trusted hardware, there are more chances of key leakaging. The flexible key delegation could be an interesting direction for designing leakage resilient cryptosystem.

Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar.

## VI. REFERENCES

[1] Cheng-Kang Chu, Sherman S. M. Chow, Wen-Guey Tzeng, Jianying Zhou, and Robert H. Deng ,Senior Member, IEEE, ”Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage”, IEEE, 2013.

[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]S.Surya, V.Karuppuchamy, ”Secure Sharing Of Data for Dynamic Multi Owner in Cloud Storage”, International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE),Vol.2, Special Issue 1, March 2014.

[4] Danan Thilakanathan, Shiping Chen, Surya Nepal and Rafael A. Calvo, “Secure Data Sharing in the Cloud”,Springer 2014.

[5] Boyang Wang, Sherman S. M. Chow, Ming Li, and Hui Li,“Storing Shared Data on the Cloud via Security-Mediator”, ICDCS 2013.

[6]C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, “Privacy-Preserving Public Auditing for Secure Cloud Storage,” IEEE Trans. Computers, vol. 62, no. 2, pp. 362–375, 2013.

[7]R. S. Sandhu, “Cryptographic Implementation of a Tree Hierarchy for Access Control,” Information Processing Letters, vol. 27, no. 2,pp. 95–98, 1988.

[8]M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, “Dynamic and Efficient Key Management for Access Hierarchies,” ACM Transactions on Information and System Security (TISSEC), vol. 12,no. 3, 2009.

[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 (Encrypt ’07), ser. LNCS, vol. 4990.Springer, 2007, pp. 384–398.

[10] 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.

[11]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.