# CP-ABE Based Encryption for Secured Cloud Storage Access

B. Raja Sekhar,B. Sunil Kumar, L. Swathi Reddy, V. PoornaChandar

**Abstract** ---Database outsourcing to cloud storage servers is emerging trend among many firms and users. Cloud server stores the user data at various host Data centers, which are remotely located. Manyorganizations outsource their data management needs to an external service provider to relief the burden of storage management, universal data access with independent geographical locations, and avoidance of capital expenditure on hardware, software, personnel maintenance. While cloud computing makes these advantages more appealing than ever, it also 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. To ensure the data security and integrity in cloudstorage, until now, a large number of techniques have been proposed to address this problem, but all of them have their own inherent limitations. In this paper,Ciphertext policy attribute-based encryption (CP-ABE) is introduced 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. This effectively eliminates the need to rely on the data storage server for preventing unauthorized data access and integrity. The performance and security analyses indicate that the proposed scheme is efficient to securely manage the data stored in the data storage servers.

**Keywords** --- Cloud storage,CP-ABE,Data Security,KGC,Access tree, Outsourcing,cloud storage servers,Fine grained access.

## 1. INTRODUCTION

Cloud storage is a model of networked online storage where data is stored in virtualized pools of storage which are generally hosted by third parties. Hosting companies operate large data centers, and people who require their data to be hosted buy or lease storage capacity from them. The data center operators, in the background, virtualize the resources according to the requirements of the customer and expose them as storage pools, which the customers can themselves use to store files or data objects. Physically, the resource may span across multiple servers. By outsourcing, organizations can concentrate on their core tasks and operate other business applications via the Internet, rather than incurring substantial hardware, software and personnel costs involved in maintaining applications in house. As data owners no longer physically possess the storage of their data, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted [6, 7]. In particular, simply downloading the data for its integrity verification is not a practical solution due to the high cost of input/output (I/O) and transmission across the network.W As people enjoy the advantages of these new technologies and services, their concerns about data security and access control also arise. Improper use of the data by the storage server or unauthorized access by outside users could be potential threats to their data. People would like to make their sensitive or private data only accessible to the authorized people with credentials they specified.

Attribute-based encryption (ABE) is a promising cryptographic approach that achieves a fine-grained data access control. [3]. It provides a way of defining access policies based on different attributes of the requester, environment, or the data object. Especially, ciphertext-policy attribute based encryption (CP-ABE) enables an encryptor to define the attribute set over a universe of attributes that a decryptor needs to possess in order to decrypt the ciphertext, and enforce it on the contents [5]. Thus, each user with a different set of attributes is allowed to decrypt different pieces of data per the security policy. This effectively eliminates the need to rely on the data storage server for preventing unauthorized data access, which is the traditional access control approach of such as the reference monitor [1].

## 2. RELATED WORK

To ensure the data in storage servers, previously the simplest Proof of retrievability (POR) scheme[6] can be made using a keyed hash function hk(F). In this scheme the verifier, before archiving the data file F in the cloud storage, pre-computes the cryptographic hash of F using hk(F) and stores this hash as well as the secret key K. To check if the integrity of the file F is lost the verifier releases the secret key K to the cloud archive and asks it to compute and return the value of hk(F). Though this scheme is very simple and easily implementable the main drawback of this scheme are the high resource costs it requires for the implementation discussed in [7]. At the verifier side this involves storing as many keys as the number of checks it want to perform as well as the hash value of the data file.

Since the introduction of ABE in implementing fine-grained access control systems, a lot of works have been proposed to design flexible ABE schemes. There are two methods to realize the fine-grained access control based on ABE: KP-ABE and CP-ABE. They were both mentioned in [4] by Goyal et al. In KP-ABE, each attribute private key is associated with an access structure that specifies which type of ciphertexts the key is able to decrypt, and ciphertext is labeled with sets of attributes. In a CP-ABE system, a user's key is associated with a set of attributes and an encrypted ciphertext will specify an access policy over attributes. The first KP-ABE construction [4]realized the monotonic access structures for key policies. Bethencourt et al. [2] proposed the first CP-ABE construction. The construction [2] is only proved secure under the generic group model. To overcome this weakness, Cheung and Newport [3] presented another construction that is proved to be secure under the standard model. To achieve receiver-anonymity, Boneh and Waters [10] proposed a predicate encryption scheme based on the primitive called Hidden Vector Encryption. In this proposed scheme data owners need not be concerned about defining any access policy for users, but just need to define only the access policy for attributes as in the previous ABE schemes.

## 3. CP-ABE BASED SECURED CLOUD STORAGE ARCHITECTURE

In this section, first, we give a formal definition of our proposed scheme, and later we give the security model in which our scheme is proven to be secure.

### 3.1. System Description

Fig. 1 shows the architecture of the data sharing system, which consists of the following system entities:

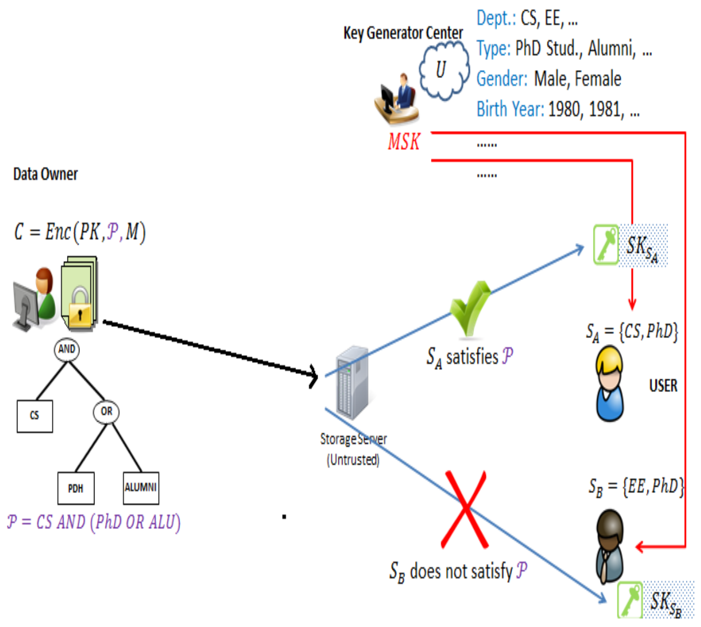
Key Generator Center
U
Dept.: CS, EE, ...
Type: PhD Stud., Alumni, ...
Gender: Male, Female
Birth Year: 1980, 1981, ...
MSK
......
......
Data Owner
$C = Enc(PK, \mathcal{P}, M)$
AND
CS
OR
PDH
ALUMNI
$\mathcal{P} = CS\ AND\ (PhD\ OR\ ALU)$
Storage Server (Untrusted)
$S_A$ satisfies $\mathcal{P}$
$SK_{S_A}$
$S_A = \{CS, PhD\}$
USER
$S_B = \{EE, PhD\}$
$S_B$ does not satisfy $\mathcal{P}$
$SK_{S_B}$

*Figure 1. CP-ABE based Storage Architecture*

***Key generation center:*** It is a key authority that generates public and secret parameters for CP-ABE. It is in charge of issuing, revoking, and updating attribute keys for users. It grants differential access rights to individual users based on their attributes. It is assumed to be honest but curious. That is, it will honestly execute the assigned tasks in the system; however, it would like to learn information of encrypted contents as much as possible. Thus, it should be prevented from accessing the plaintext of the encrypted data even if it is honest.

***Data storing center:*** It is an entity that provides a data sharing service. It is in charge of controlling the accesses from outside users to the storing data and providing corresponding contents services. The data storing center is another key authority that generates personalized user key with the KGC, and issues and revokes attribute group keys to valid users per each attribute, which are used to enforce a fine-grained user access control. Similar to the previous schemes [2],[3],[4], we assume the data storing center is also semi-trusted (that is, honest-but-curious) like the KGC.

***Data owner:*** It is a client who owns data, and wishes to upload it into the external data storing center for ease of sharing or for cost saving. A data owner is responsible for defining (attribute- based) access policy, and enforcing it on its own data by encrypting the data under the policy before distributing it.

***User:*** It is an entity who wants to access the data. If a user possesses a set of attributes satisfying the access policy of the encrypted data, and is not revoked in any of the valid attribute groups, then he will be able to decrypt the ciphertext and obtain the data.

## 3.2.Ciphertext-Policy Attribute-Based Encryption and Access Format

In our construction private keys will be identified with a set of descriptive attributes. A party that wishes to encrypt a message will specify through an access tree structure a policy that private keys must satisfy in order to decrypt. Each interior node of the tree is a threshold gate and the leaves are associated with attributes. A user will be able to decrypt a ciphertext with a given key if and only if there is an assignment of attributes from the private key to nodes of the tree such that the tree is satisfied. We use the same notation as[5] to describe the access trees ,even though in our case the attribute s are used to identify the keys(as opposed to the data).specified in the private key, while the ciphertexts are simply labeled with a set of descriptive.

*Access tree T structure :* Let T be a tree representing an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. If num x is the number of children of a node x and k, x is its threshold value, then $0 < k_x = num_x$. When $k_x = 1$, the threshold gate is an OR gate and when $k_x = num_x$, it is an AND gate. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$.

To facilitate working with the access trees, we define a few functions. We denote the parent of the node x in the tree by parent(x). The function att(x) is defined only if x is a leaf node and denotes the attribute associated with the leaf node x in the tree. The access tree T also defines an ordering between the children of every node, that is, the children of a node are numbered from 1 to num. The function index(x) returns such a number associated with the node x.Where the index values are uniquely assigned to nodes in the access structure for a given key in an arbitrary manner.

**Satisfying an access tree :**Let T be an access tree with root r. Denote by $T_x$ the sub tree of T rooted at the node x. Hence T is the same as $T_r$. If a set of attributes Y satisfies the access tree $T_x$, we denote it as $T_x(Y) = 1$. We compute $T_x(Y)$ recursively as follows. If x is a non-leaf node, evaluate $T_x$ (Y) for all children x of node x. $T_x(Y)$ returns 1 if and only if at least $k_x$ children return 1. If x is a leaf node, then $T_x(Y)$ returns 1 if and only if att(x) Є Y.

**Encrypt(PK,M,T ) :**The encryption algorithm encrypts a message M under the tree access structure T . The algorithm first chooses a polynomial qx for each node x (including the leaves) in the tree T . These polynomials are chosen in the following way in a top-down manner, starting from the root node R. For each node x in the tree, set the degree dx of the polynomial qx to be one less than the threshold value $k_x$ of that node, that is,$d_x = k_x -1$. Starting with the root node R the algorithm chooses a random s ? p and sets qR(0) = s. Then, it chooses Z dR other points of the polynomial qR randomly to define it completely. For any other node x, it sets qx(0) =qparent(x)(index(x)) and chooses dx other points randomly to completely define qx. Let, Y be the set of leaf nodes in T . The ciphertext is then constructed by giving the tree access structure T and computing

$$CT = (T , C = Me(g,g)^{\alpha s} ,\ C = h^{s} ,$$

$$\forall_{y} \in Y : Cy = g^{qy} ,C_{y} = H(att(y))^{qy(0)} ).$$

**KeyGen(MK,S)**. The key generation algorithm will take as input a set of attributes S and output a key that identifies with that set The algorithm first chooses a random r ? ,Z p and then random r ? Z for each attribute j ? S. Then it j p computes the key

$$SK = (D = g^{(\alpha+r)/\beta},$$
$$\forall j \in S : D_{j}= g^{r\cdot} H(j)^{rj}, D_{j}= g^{rj}).$$

**Delegate(SK,S).** The delegation algorithm takes in a secret key SK, which is for a set S of attributes, and another set $S^1$ such that $S^1$ Є S. The secret key is of the form SK = (D, $\forall j \in S$ : $D_j,D^1{}_j$ ). Then algorithm chooses random the r˜and r $_{\tilde{}k}$ ∀ k Є S`. Then it creates a new secret key as

$$SK^{\sim}=(D^{\sim}= Df^{r}, \forall k\in S^{\sim}: D_{k}= D_{k}g^{r}H(k)^{rk},D^{\sim}{}_{k}= D^{-}{}_{k}g^{rk})$$

The resulting secret key SK is a secret key for the set S. Since the algorithm re-randomizes the key, delegated key is equivalent to one received directly from the authority.

**Decrypt(CT,SK)**. We specify our decryption procedure as a recursive algorithm. For ease of exposition we present the simplest form of the decryption algorithm and discuss potential performance improvements in the next subsection. We first define a recursive algorithm Decrypt Node(CT,SK,x) that takes as input a ciphertext

$$\tilde{CT} = (T, \tilde{C}, C, \forall y \in Y : C_y, C'_y)$$

a private key SK, which is associated with a set S of attributes, and a node x from T . If the node x is a leaf node then we let i = att(x) and define as follows: If i ∈ S, then leaf node algorithm DecryptNode(CT,SK,x) proceeds as follows: For all nodes z that are children of x, it calls DecryptNode(CT,SK,z) and stores the output as $F_z$. Let $S_x$ be an arbitrary $k_x$- sized set of child nodes z.such that $F_z \neq \pm$ . If no such set exists then the node was not satisfied and the function returns ±.If i ≠ S, then we define DecryptNode(CT,SK,x) = ± .

## 4. CONCLUSIONS

We created a storage access policy system for Ciphertext-Policy Attribute Based Encryption. Our system allows for a new type of encrypted access control where user's private keys are specified by a set of attributes and a party encrypting data can specify a policy over these attributes specifying which users are able to decrypt. Our system allows policies to be expressed as any monotonic tree access structure and is resistant to collusion attacks in which an attacker might obtain multiple private keys. Finally, we provided an implementation of our system, which included several optimization techniques and involved data owner in secured data security policies definition.

The KGC and the data storing center are involved in the user key issuing protocol as explained above. In this protocol, a user is required to contact the two parties before getting a set of keys. The KGC is responsible for authenticating a user and issuing attribute keys to him if the user is entitled to the attributes. The secret key is generated through the secure 2PC protocol between the KGC and the data storing center. They engage in the arithmetic secure 2PC protocol with master secret keys of their own, and issue independent key components to a user. Then, the user is able to generate the whole secret keys with the key components separately received from the two authorities. The secure 2PC protocol deters them from knowing each other's master secrets so that none of them can generate the whole secret keys of a user alone.

The above Fine-grained access control systems facilitate granting differential access rights to various users and allow flexibility in specifying the access rights of individual users.

## 5. REFERENCES

[1] [SW05] Sahai, A., Waters, B.: Fuzzy identity-based encryption. EUROCRYPT 2005.

[2] [GPSW06] Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for finegrained access control of encrypted data. ACM CCS 2006.

[3] [BSW07] Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. IEEE Symposium on Security and Privacy, 2007

[4] [CN07] Cheung, L., Newport, C.C.: Provably secure ciphertext policy abe. ACM CCS 2007

[5] [GJPS08] Goyal, V., Jain, A., Pandey, O., Sahai, A.: Bounded Ciphertext Policy Attribute Based Encryption. ICALP 2008, Part II.

[6] [Waters08/11] Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. PKC 2011

[7] [LOSTW10]Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (Hierarchical) inner product encryption. EUROCRYPT 2010.

[8] [OT10] Okamoto, T., Takashima, K. : Fully secure functional encryption with general relations from the decisional linear assumption. CRYPTO 2010.

[9] [MKE09] Muller, S., Katzenbeisser, S., Eckert, C.: On multi-authority ciphetext-policy attribute-based encryption. Bulletin of the Korean Mathematical Society 2009.

[10] [LW11] Lewko, A., Waters, B.: Decentralizing attribute-based encryption. EUROCRYPT 2011.

Mr. **B. RAJA SEKHAR** working as Asst Prof in CSE Department in Jawaharlal Nehru Institute of Technology, Hyderabad. He completed his B.Tech(CSE) and M.Tech(SE) from JNTU Hyd. He is having 4years of Teaching Experience and member of IACSIT and CSTA. His interested subjects are Data Mining, Algorithms, DS through C, Software Engineering, Image Processing. rajsekhar.cse@gmail.com

Mr. **B. SUNIL KUMAR** working as Asst Prof in MCA Department in Jawaharlal Nehru Institute of Technology, Hyderabad. He completed his MCA from Osmania University and Pursuing M.Tech(CSE) from JNTU Hyderabad. He is having

3years of Teaching Experience and interested subjects are Data Mining, Web Technologies, Web Services, JAVA, C, Operating Systems. sunilkumar0060@gmail.com

Ms. **L. SWATHI REDDY** working as Asst Prof in CSE Department in Jawaharlal Nehru Institute of Technology, Hyderabad. She completed herB.Tech(CSE) from JNTU Hyderabad. Her interested subjects are Data Mining, Data Structures using C, Software Engineering, Operating Systems. lokasaniswathireddy@gmail.com

**V. POORNA CHANDAR, MCA** working as an Asst Prof in MCA Department for Guru Nanak Institute of PG Studies, Hyderabad, He had 5 years of teaching Experience. He completed MCA through Kakatiya University, Warangal and his interested subjects are C, C++, Java, Web Technologies, Network and Information Security, Data Mining,Software analysis and Design Engineering. **poorimcaonline@gmail.com**