ISSN 2348 – 9928
IJAICT Volume 5, Issue 12, December 2018 Doi:01.0401/ijaict.2015.04.01 Published on 05 (12) 2018

PRIVACY ENHANCED DATA ACCESS CONTROL FOR USERS OVER MULTI-AUTHORITY CLOUD STORAGE

Mr. S. Ayyasamy
Assistant Professor,
Sri Eshwar College of Engineering,
Coimbatore, Tamilnadu, India

Mr. S. T. Balaji
Chief Operating Officer,
Sarobal Soft Solutions,
Kasaragod, Kerala, India

Mr. K. Senthil Kumar
Assistant Professor,
Sri Eshwar College of Engineering,
Coimbatore, Tamilnadu, India

Abstract: The number of user in cloud computing are increasing tremendously due to its advantage of providing flexible storage requirement. The users are started to share their sensitive information through the cloud due to its nature of user friendly environment. The security of the data has to be assured to the users when storing their data's into the cloud server. In the existing work an expressive, efficient and revocable data access control scheme for multi-authority cloud storage systems is proposed to support the access control by using attributes gathered from multiple authorities. The users those who are having matching attributes as in the access policy defined in the cipher text can retrieve the entire data content. It aims to allow the users with eligible attributes to decrypt the entire data stored in the cloud server. However it cannot provide the promisable data access control to the users due to the presence of malicious attribute authorities. In the case of malicious attribute authority, the data security may be violated. In our work a novel approach namely Attribute Authority Revocation based trapdoor commitment scheme is introduced. This algorithm provides a better way to find out the malicious attribute authorities by making a commitment before exchanging their information.

Keywords— Attributes, Cipher text policy, Access control.

I. INTRODUCTION

Cloud computing is a promising computing paradigm which recently has drawn extensive attention from both academia and industry. By combining a set of existing and new techniques from research areas such as Service-Oriented Architectures (SOA) and virtualization, cloud computing is regarded as such a computing paradigm in which resources in the computing infrastructure are provided as services over the Internet. Along with this new paradigm, various business models are developed, which can be described by terminology of "X as a service (XaaS)" where X could be software, hardware, data storage, and etc. Successful examples are Amazon's EC2 and S3, Google App Engine, and Microsoft Azure which provide users with scalable resources in the pay-as-you use fashion at relatively low prices. For example, Amazon's S3 data storage service just charges $12 to $15 per gigabyte month. As compared to building their own infrastructures, users are able to save their investments significantly by migrating businesses into the cloud. With the increasing development of cloud computing technologies, it is not hard to imagine that in the near future more and more businesses will be moved into the cloud.

Cloud computing is also facing many challenges that, if not well resolved, may impede its fast growth. Data security, as it exists in many other applications, is among these challenges that would raise great concerns from users when they store sensitive information on cloud servers. These concerns originate from the fact that cloud servers are usually operated by commercial providers which are very likely to be outside of the trusted domain of the users. Data confidential against cloud servers is hence frequently desired when users outsource data for storage in the cloud. Access control is the selective restriction of access to a place or other resource. The act of accessing may mean consuming, entering, or using. Permission to access a resource is called authorization [2]. Locks and login credentials are two analogous mechanisms of access control. Data access control in the cloud leads to an security concerns which is provide an main research goal to the researchers and developer of cloud computing [8]. The main goal of our project is to provide a privacy and security concern for the cloud storage data owner when they are outsourcing their confidential data to the public cloud.

The main objective of our work is to provide the prevention from the malicious attribute authorities who may affect the performance of the users where there is no central authority to control the functionality of the multiple authorities [7]. Generally the central authority which is responsible to integrate the secret keys from the other attribute authorities, and thus to integrate these secret keys without the central authority would be an obstacle in our work.

II. SYSTEM MODEL

We consider a data access control system in multi-authority cloud storage, as described in Fig. 1. There are five types of entities in the system: a certificate authority (CA), attribute authorities (AAs), data owners (owners), the cloud server (server) and data consumers (users).

The CA is a global trusted certificate authority in the system [1]. It sets up the system and accepts the registration of all the users and AAs in the system. For each legal user in the system, the CA assigns a global unique user identity to it and also generates a global public key for this user. However, the CA is not involved in any attribute management and the creation of secret keys that are associated with attributes.

Every AA is an independent attribute authority that is responsible for entitling and revoking user's attributes according to their role or identity in its domain. In our scheme, every attribute is associated with a single AA, but each AA can manage an arbitrary number of attributes. Every AA has full control over the structure and semantics of its attributes. Each AA is responsible for generating a public attribute key for each attribute it manages and a secret key for each user reflecting his/her attributes.

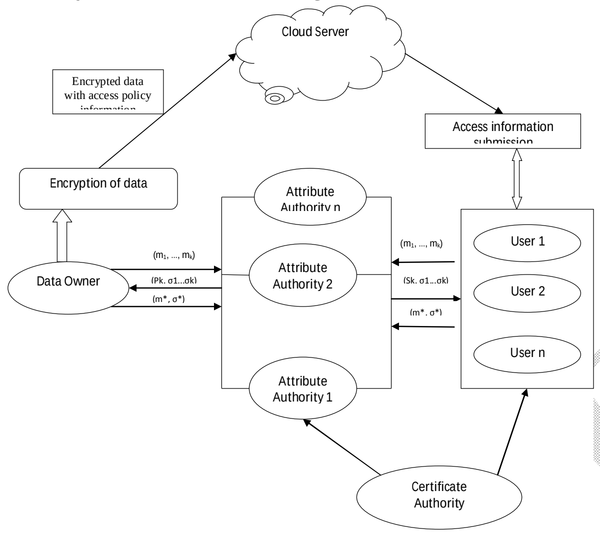

Fig 1. System Model for Multi Authority Cloud Storage

Each user has a global identity in the system. A user may be entitled a set of attributes which may come from multiple attribute authorities [4]. The user will receive a secret key associated with its attributes entitled by the corresponding attribute authorities [5].

Each owner first divides the data into several components according to the logic granularities and encrypts each data component with different content keys by using symmetric encryption techniques. Then, the owner defines the access policies over attributes from multiple attribute authorities and encrypts the content keys under the policies. Then, the owner sends the encrypted data to the cloud server together with the ciphertexts[2]. They do not rely on the server to do data access control. But, the access control happens inside the cryptography. That is only when the user's attributes satisfy the access policy defined in the ciphertext, the user is able to decrypt the ciphertext. Thus, users with different attributes can decrypt different number of content keys and thus obtain different granularities of information from the same data.

In our work we consider the non interactive trapdoor commitment scheme. A trapdoor is inserted between the AA's and the users as well as between the AA's and the data owners in order to provide security to the data by eliminating the malicious AA's. A commitment scheme is a primitive to generate and open commitments. More precisely a commitment scheme is a two-phase protocol between two probabilistic polynomial time algorithms sender and receiver. In a first stage (called the commitment phase) sender commits to a message m using some appropriate function Com which takes as input m and some auxiliary value r and produces as output a value c. The value c is sent to receiver as a commitment on m. In the second stage (called the de-commitment phase) sender "convinces" receiver that c is actually a valid commitment on m (if receiver is not convinced, it just outputs some special string).

III. FRAMEWORK

A. *System Initialization:*

This phase consists of CA setup and AA setup with the following algorithms:

CASetup (1^{λ}): (GMK, GPP, (GPK'_{uid}, GPK'_{uid}), (GSK_{uid};GSK'_{uid}), Certificate (uid)).

A certificate authority (CA) is an authority in a network that issues and manages security credentials and public keys for message encryption. As part of a public key infrastructure (PKI), a CA checks with a registration authority (RA) to verify information provided by the requestor of a digital certificate. If the RA verifies the requestor's information, the CA can then issue a certificate.

Depending on the public key infrastructure implementation, the certificate includes the owner's public key, the expiration date of the certificate, the owner's name, and other information about the public key owner.

The CA setup algorithm is run by the CA.

- It takes no input other than the implicit security parameter λ.
- It generates the global master key GMK of the system and the global public parameters GPP.
- For each user uid, it generates the user's global public keys (GPKuid, GPK'uid), the user's global secret keys (GSKuid, GSK'uid) and a certificate {Certificate (uid)} of the user.

AASetup (U_{aid}): (SK_{aid}, PK_{aid}, $\{VK_{xaid}, PK_{xaid}\}_{xaid}$, U_{aid})

An attribute authority generates a public attribute key for each attribute it maintains; this public key is available to every users and data owners. Furthermore, the attribute authorities determine eligible users and distribute personalized secret attribute keys over an authenticated and trusted channel to them.

The attribute authority setup algorithm is run by each attribute authority.

- It takes the attribute universe U_{aid} managed by the AA_{aid} as input.

- It outputs a secret and public key pair (SKaid, PKaid) of the AA_{aid} and a set of version keys and public attribute keys $\{VK_{xaid}, PK_{xaid}\}_{xaid}$ □ U_{aid} for all the attributes managed by the AA_{aid}.

B. Implementation of Data Access Control Scheme

Secret key distribution

Attribute authority will generate a secret key for an individual authorized user in order to provide them the efficient data access control. The secret will be generated by using the individual user's unique identity key and the attributes defied for the users. In order to obtain the corresponding secret keys, a user must submit his GID to each authority. So, multiple authorities can cooperate to collect the user's attributes by it. The secret key generation process is defined below:

- A randomized algorithm takes as input the authority's secret key SK, a user u's UID, and a set of attributes A_{ku} in the authority AA_k's domain (We will assume that the user's claim of these attributes has been verified before this algorithm is run, $A_u = \{A_{ku},\ k = 1, \ldots, n\}$).
- Output a secret key Du for the user u.

Attribute distribution to data owner

Once the attributes for the users presents in the network are defined and generated the unique secret key, the list of attributes will be distributed to the data owner by using which he will encrypt the data's. That's the collected attributes from all attribute authorities (AC) will be sent to the users for the encryption purpose. By using these attributes defined by the attribute authorities, the access police will be generated in order to limit the data accessing of multiple users. The user who matches with the access policy only can decrypt the data.

Data Encryption

A randomized algorithm takes as input an attribute set AC of a message M, the system public parameters PK and outputs the cipher text C.

Encryption is done as follows:

- Choose a random value s □ Zq. For the attribute set Ac = { A_c^k }, k □ {1, … , n}, generate the cipher text C = {Ac, E = $e(g1,g2)^s$. M||$O^{l'}$, {$E_{k,j}$ = $T_{k,j}^s$ }j□ A_c^k for all k} Where k corresponds to the authority AAl, A_c^k denotes the attribute set of the cipher text monitored by authority AA_k.
- In order to check whether a decryption is valid, prior to encryption, we append M trailing 0s denoted 0l' to message M □ $\{0,1\}^l$

Data Decryption

Users first run the decryption algorithm to get the content keys, and use them to further decrypt the data.

$$\text{Decrypt}(CT, GPK_{uid}, GSK'_{uid}, \{SK_{uid,aidk}\}_{aidk \square IA}) \rightarrow k.$$

The decryption algorithm is run by users to decrypt the cipher text. It takes as inputs the cipher text CT which contains an access policy A, a global public key GPK_{uid} and a global secret key GSK'_{uid} of the user *uid*, and a set of secret keys $\{SK_{uid,aidk}\}_{aidk}$□I_A from all the involved AAs. If the attributes $\{S_{uid,aidk}\}_{aidk}$□I_A of the user uid satisfy the access policy A, the algorithm will decrypt the cipher text and return the content key k.

C. User Revocation

In this module, user revocation is handled by eliminating the revoked user's key by updating it. Suppose an attribute $x_{aid'}$ is revoked from the user uid' by the $AA_{aid'}$. The attribute $x_{aid'}$ is denoted as the Revoked Attribute and the user uid' is denoted as the Revoked User. We also use the term of Non-revoked Users to denote the set of users who possess the revoked attribute $x_{aid'}$ but have not been revoked. This is done in three steps that can be as follows.

Update Key Generation

When an attribute xaid' is revoked from a user, the corresponding authority AAaid' runs the update key generation algorithm UKeyGen to compute the update keys. The algorithm takes as inputs the secret key $SK_{aid'}$ of $AA_{aid'}$, the revoked attribute $x_{aid'}$ and its current version key $VK'_{xaid'}$. It generates a new version key $VK'_{xaid'} = v'_{xaid'}$ ($v'_{xaid'}! = v_{xaid'}$) for the revoked attribute $x_{aid'}$. The $AA_{aid'}$ then generates a unique update key $UK_{s,\ xaid,\ uid}$ for secret key update by each non revoked user uid as

$$UK_{s,xaid,uid} = H(x_{aid'})^{\beta aid\ (v'xaid,\ -vxaid')(uuid+\gamma aid')}$$

And generates the update key $UK_{c,xaid}$ for cipher text update. The AA_{aid} sends the $UK_{s,xaid',uid}$ to non-revoked user uid and sends $UK_{c,xaid'}$ to the cloud server.

Secret key update by Non revoked users

Upon receiving the update key $UK_{s,\ xaid',\ uid}$, the user uid then update his/her secret key by running the new secret key update algorithm SKUpdate.

Cipher Text Update by Server

Upon receiving the update key $UK_{c,xaid'}$ from the authority. The cloud server runs the cipher text update algorithm CTUpdate to update the cipher text associated with the revoked attribute $x_{aid'}$. It takes as inputs the cipher texts associated with the revoked attribute $x_{aid'}$ and the update key $UK_{c,xaid'}$. It updates the cipher texts that are associated with the revoked attribute $x_{aid'}$

D. AA Revocation with trapdoor

In this module, Attribute Authority revocation is done based on the trapdoor commitment scheme, so that the unauthorized Attribute Authorities can get revoked. This trapdoor commitment scheme is run as follows:

Input: Message space $M_{ck} = G_2^n$, randomizer space $R_{ck} = G_2$ and commitment space $C_{ck} = G_T$
Setup: On input 1^k return gk = (p, G1, G2, GT, e) ← g (1^k) // *secret parameter key*
Key Generator: On input gk pick at random g_r← G1 \ {1} and x1, …., xn ← Zp and define $g1 = g_r^{x1}, \ldots g_n = g_r^{xn}$. The commitment key id ck = (gk, gr, g1, …, gn) and the trapdoor key is tk = (gk, x1, …, xn)
Commitment: Using commitment key ck on input message (m1,…mn) G_2^n pick randomizer r ← G2. The commitment is given by

$$C = e\,(gr, r) = \prod_{i=1}^{n} e(gi, mi)$$

Trapdoor Commitment: Using commitment key ck and trapdoor key tk generate an equivocal commitment c GT by picking r G2 and computing c = e (gr, r). The corresponding equivocation key is ek = (tk, r).
Trapdoor opening: On an equivocal commitment c GT to a message (m1, …, mn) G_2^n using the equivocation key ek, compute and return the trapdoor opening r' = r.

$$\prod_{i=1}^{n} m_i^{-xi}$$

IV. Security Analysis

A. Backward Security

During the secret key updating, the corresponding AA generates an update key for each non-revoked user. Because the update key is associated with the user's global identity uid, the revoked user cannot use update keys of other non-revoked users to update its own secret key, even if it can compromise some non-revoked users [3]. This guarantees the backward security.

B. Forward Security

After each attribute revocation operation, the version of the revoked attribute will be updated. When new users join the system, their secret keys are associated with attributes with the latest version [6]. However, previously published cipher texts are encrypted under attributes with old version. The cipher text update algorithm in our protocol can update previously published cipher texts into the latest attribute version, such that newly joined users can still decrypt previously published cipher texts, if their attributes can satisfy access policies associated with cipher texts. This guarantees the forward security.

V. Performance Analysis

A. Success Rate

When Compared to the existing model [1], the rate of success for acquiring the data from the cloud server by the users are improved. Because of the AA revocation, the malicious AA's are eliminated so that the correct secret keys and public keys will be issued to the users and data owners. So the numbers of wrong keys given by the AA's are eliminated so that the success rate for accessing the data is improved.

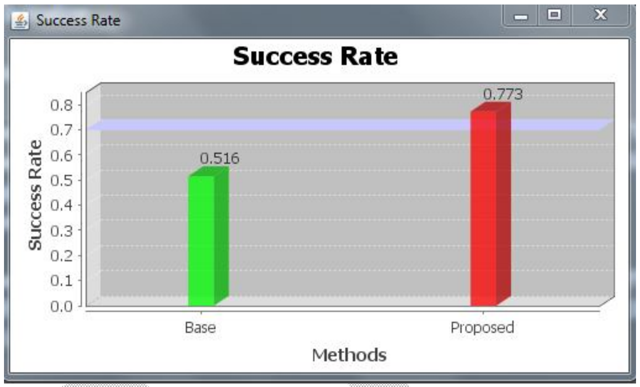

Fig 2 : Success Rate

In the fig 2, success rate taken to process the user submitted requests for both the existing and the proposed methodology is measured. In the x axis methods are defined and in y axis success rate is depicted. From this graph, it can be proved that the success rate taken to process the user requests is increased considerably in the proposed approach than the existing method.

B. Storage Cost

The storage cost is defined as the total amount which is paid by the users for storing their personal contents. The storage cost of the proposed methodology need to be reduced in order to improve the resource utilization and as well as improve the user satisfaction level. The comparison graph is depicted as in the following graph

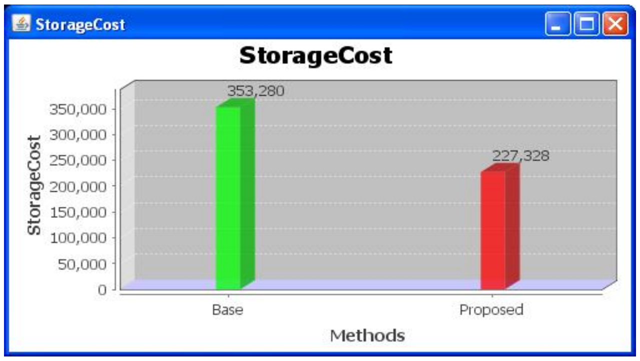

Fig 3 :Storage Cost

In the fig 3 storage cost taken to process the user submitted requests for both the existing and the proposed methodology is measured. In the x axis methods are defined and in y axis storage cost is depicted. From this graph, it can be proved that the storage cost taken to process the user requests are reduced considerably in the proposed approach than the existing method.

ISSN 2348 – 9928
Doi:01.0401/ijaict.2015.04.01 Published on 05 (12) 2018

C. Time Comparison

The time is measured by calculating the terms of waiting time of user requests and the time taken to process them. The total time need to be reduced in the proposed methodology where the number of requests that are handled can be improved considerably within less period of time. The comparison graph is depicted in the following graph.

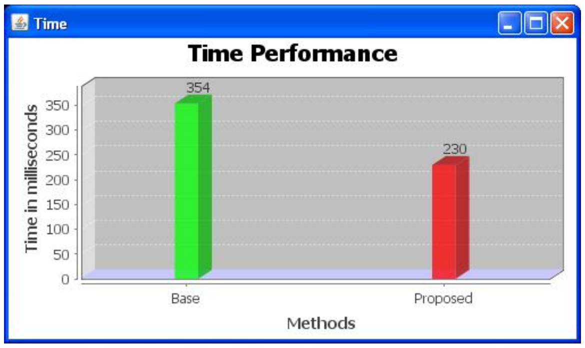

Fig 4 : Time Performance

In the fig 4,time taken to process the user submitted requests for both the existing and the proposed methodology is measured. In the x axis methods are defined and in y axis time taken to process the user requests are depicted. From this graph, it can be proved that the total time taken to process the user requests is reduced considerably in the proposed approach than the existing method.

VI. Conclusion

An Attribute Authority Revocation based trapdoor commitment scheme is proposed to provide an efficient access control over a third party providers and users with security concern. The malicious users and malicious providers cannot access the data without knowing the privileged commitment message details. The full-fledged implementation of the mechanism on commercial public cloud as an important future extension, which is expected to robustly cope with very large scale data and thus encourage users to adopt cloud storage services more confidently.

References

[1] Kan Yang, Xiaohua Jia (2014), "Expressive, Efficient, and Revocable Data Access Control for Multi-Authority Cloud Storage".

[2] Brent Waters (2011), "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization", Public key cryptography, Lecture Notes in Computer Science Volume 6571, pp 53-70

[3] S. Yu, C. Wang, K. Ren, and W. Lou, ''Attribute Based Data Sharing with Attribute Revocation,'' in Proc. 5th ACM Symp. Information, Computer and Comm. Security (ASIACCS'10), 2010, pp. 261-270.

[4] M. Chase, ''Multi-Authority Attribute Based Encryption,'' in Proc. 4th Theory of Cryptography Conf. Theory of Cryptography (TCC'07), 2007, pp. 515-534.

[5] M. Chase and S.S.M. Chow, ''Improving Privacy and Security in Multi-Authority Attribute-Based Encryption,'' in Proc. 16th ACM Conf. Computer and Comm. Security (CCS'09), 2009, pp. 121-130.

[6] J. Hur and D.K. Noh, ''Attribute-Based Access Control with Efficient Revocation in Data Outsourcing Systems,'' IEEE Trans. Parallel Distributed Systems, vol. 22, no. 7, pp. 1214-1221, July 2011.

[7] K. Yang and X. Jia, ''Attribute-Based Access Control for Multi-Authority Systems in Cloud Storage,'' in Proc. 32th IEEE Int'l Conf. Distributed Computing Systems (ICDCS'12), 2012, pp. 1-10.

[8] J. Bethencourt, A. Sahai, and B. Waters, ''Ciphertext-Policy Attribute-Based Encryption,'' in Proc. IEEE Symp. Security and privacy (S&P'07), 2007, pp. 321-334..