Secured Anonymous Authentication of Data with Decentralized Access Control in Cloud

Manjunath R
PG student, Department of CS&E,
A.P.S College of Engineering,
Bangalore, India.

Prof Bhagyashree R
Assistant Professor, Department of CS&E,
A.P.S College of Engineering,
Bangalore, India

***Abstract*— Cloud Computing is a multi-tenancy feature, which provides privacy, security and access control challenges and uses the internet and central remote servers to maintain data and applications. In order to achieve safe storage, policy based file access control and policy based file assured deletion of a file stored in a cloud environment, a suitable encryption technique with key management should be applied before outsourcing the data. In this work, the proposed new DAC scheme allows secure data storage in clouds that supports anonymous authentication. We implemented secure cloud storage by providing access to the files with the policy based file access using Attribute Based Encryption (ABE) scheme with RSA key public-private key combination. Private Key is the combination of the user's credentials. So that high security will be achieved. When the time limit of the file expired, the file will be automatically revoked and cannot be accessible to anyone in future. Policy based file renewal is proposed. The Renewal can be done by providing the new key to the existing file, will remains the file until the new time limit reaches.**

***Index Terms*—Access control, authentication, Decentralized access, attribute-based encryption, access policy, claim predicate, cloud storage.**

I. INTRODUCTION

In cloud computing, users can outsource their computation and storage to servers (also called clouds) using Internet. This frees users from the hassles of maintaining resources on-site. Clouds can
provide several types of services like applications, infrastructures and Platforms to help developers write applications much of the data stored in clouds is highly sensitive, for example, medical records and social networks. Security and User privacy are, thus, very important issues in cloud computing. In one hand, the user should authenticate itself beforeinitiating any transaction. Even cloud storage is more flexible, how the security and privacy are available for the outsourced data becomes a serious concern. There are three objectives to be main issue.
Confidentiality – preserving authorized restrictions on information access and disclosure. The main threat accomplished when storing the data with the cloud.
Integrity – guarding against improper information modification or destruction.
Availability – ensuring timely and reliable access to and useof information.

To achieve secure data transaction in cloud, suitable cryptography method is used. The data owner must encrypt the file and then store the file to the cloud. If a third person downloads the file, he/she may view the record if he/she had the key which is used to decrypt the encrypted file. Sometimes this may be failure due to the technology development and the hackers. To overcome the problem there are lot of techniques introduced to make secure transaction and secure storage. The encryption technique was implemented with set of key operations to maintain the secrecy.

Cloud servers prone to Byzantine failure, where a storage server can fail in arbitrary ways [2]. The cloud is also prone to data modification and server colluding attacks. To provide secure data storage, the data needs to be encrypted. Wang et al. [2] addressed storage security using Reed-Solomon erasure-correcting codes. Authentication of users using public keycryptographic techniques has been studied in [5].
Access control in clouds is gaining attention because it is important that only authorized users have access to valid service. There are broadly three types of access control: user-based access control (UBAC), role-based access control (RBAC), and attribute-based access control (ABAC). In UBAC, the access control list contains the list of users who are authorized to access data. This is not feasible in clouds where there are many users. In RBAC (introduced by Ferraiolo and Kuhn [8]), users are classified based on their individual roles. Data can be accessed by users who have matching roles. The roles are defined by the system. ABAC is more extended in scope, in which users are given attributes, and the data has attached access policy. Only users with valid set of attributes, satisfying the access policy, can access the data. All these work use a cryptographic primitive known as attribute based encryption (ABE). The extensible access control markup language has been proposed for ABAC in clouds [9]. An area where access control is widely being used is health care. Clouds are being used to store sensitive information about patients to enable access to medical professionals, hospital staff, researchers, and policy makers

Existing work on access control in cloud are centralized in nature [1]. The existing scheme in uses a symmetric key approach and does not support authentication. In centralized approach a single key distribution center (KDC) is responsible for distributing secret keys and attributes to all users. Unfortunately, cloud environment supports large number of users as a result, single KDC get fail in maintain all users. We, therefore, emphasize that clouds should take a

decentralized approach known as DAC scheme, while distributing secret keys and attributes to users.

II. KEY MANAGEMENT

In this paper, following are the cryptographic keys to protect data files stored on the cloud

Public Key: The Public key is a random generated binary key, generated and maintained by the Key manager itself.Particularly used for encryption/ decryption.

Private Key: It is the combination of the username, password and two security question of user's choice. Theprivate key is maintained by client itself. Used for encrypt / decrypt the file.

Access key: It is associated with a policy. Private access key is maintained by the client. The accesskey is built onattribute based encryption. File access is of read or write.

III. FORMATS OF ACCESS POLICIES

Access policies can be in any of the following formats: 1) Boolean functions of attributes, 2) linear secret sharing scheme (LSSS) matrix, or 3) monotone span programs. Any access structure can be converted into a Boolean function. An example of a Boolean function is ((a1 ^ a2 ^ a3) V (a4 ^ a5)) ^ (a6 V a7)), where a1,a2, . . . a7 are attributes.

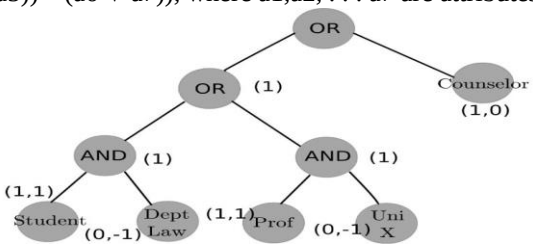


Fig. 1 structure of access policy

A. File Access Control

Ability to limit and control the access to host systems and applications via communication links. To achieve, access must be identified or authenticated. After achieved the authentication process the users must associate with correct policies with the files. To recover the file, the client must request the key manager to generate the public key. For that the client must be authenticated. The attribute based encryption standard is used for file access which is authenticated via an attribute associated with the file. With file access control the file downloaded from the cloud will be in the format of read only or write supported. Each user has associated with policies for each file. So the right user will access the right file. For making file access the attribute based encryption scheme is utilized.

IV. PROPOSED DAC SCHEME

In this section, we propose our privacy preserving authenticated access control scheme. According to our scheme a user can create a file and store it securely in the cloud. We used RSA algorithm for encryption/decryption which is an asymmetric key approach. It generates two different keys for both encryption/decryption which is not sharable. This algorithm is the proven mechanism for secure transaction. Here we are using the RSA algorithm with key size of 2048 bits.

We refer to the Fig. 2. There are three users, a creator, a reader, and writer. Creator receives a token from the trustee, who is assumed to be honest. Atrustee can be someone like head of the management who manages entire transaction. On presenting creator registration id, the trustee gives a token. A creator on presenting the token to nearest KDCs receives keys for encryption/decryption. The data or file is encrypted under the access policy *X*. The encrypted file along with time stamp is sent to the cloud. If Reader/Writer user has matching set of attributes with creator access policy then cloud allows users to decrypt and access the encrypted file based on their roles.

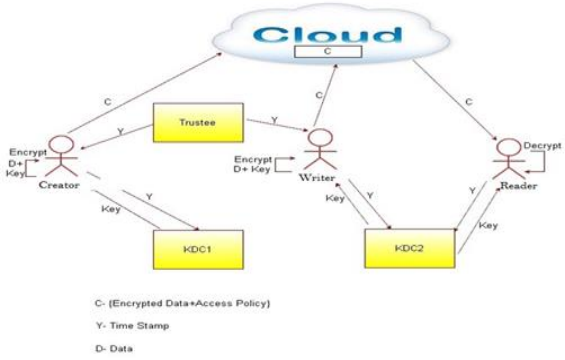


Fig 2 proposed secure cloud model

A. Attribute-Based Encryption

1. Encryption by Sender

The encryption function is **ABE:Encrypt(MSG, X).** Sender decides about the access tree X. LSSS matrix R can be derived as described in Section 3.2. Sender encrypts message MSG as follows:

1. Choose a random seed **s € Zq** and a random vector **v € Zhq** , with s as its first entry; h is the number of leaves in the access tree (equal to the number of rows in the corresponding matrix R).
2. Calculate **λ x =Rx .v**, where Rx is a row of R.
3. Choose a random vector **w € Zhq** with 0 as the first entry.
4. Calculate **wx =Rx .w.**
5. For each row Rx of R, choose a random **ρx € Zq**.
6. The following parameters are calculated:

C0 = MSGe(g,g)s,
C1,x= e(g, g)απ(x)ρx,Vx
C2,x =gρxVx,
C3,x = gyαπ(x)ρxgwxVx, (5)

where **πx** is mapping from Rx to the attribute ***i*** that is located at the corresponding leaf of the access tree.

7. The ciphertext C is sent by the sender (it also includes the access tree via R matrix):

C =(R,π ,C0,{C1,x, C2,x, C3,x, Vx}).

2. Decryption by Receiver

The decryption function is **ABE:Decrypt(C,{ski,u}),** where C is given by (5). Receiver Uu takes as input ciphertext C, secret keys {ski,u},group G0, and outputs message msg. It obtains the access matrix R and mapping _ from C. It then executes the following steps:
1.Uu calculates the set of attributes that are common to the access matrix.X is the set of rows of R.
2. For each of these attributes, it checks if there is a subset X0 of rows of R, such that the vector (1,0,. , 0) is their linear combination.If not, decryption is impossible. If yes, then it calculates constantsCx€ Zq.

B. File Upload / Download

1. File Upload

In DAC scheme, SSH protocol is used for both upload and downloading of data from cloud. On registering with trustee user will get a user id and token γ. The user receives Public keys based on token γ, for encrypting messages from KDC, then the data is encrypted under some access policy (*X)* and claim predicate (*Y),* which is a monotone Boolean function.The access policy decides who can access the data stored in the cloud and creator decides on a claim predicate *Y,* to prove data has come from right reliable source. Then the file is encrypted with the public key and private key and forwarded to the cloud.

2. File Download

The client can download the file after completion of the authentication process. User will request for token γ with the trustee using his user id. On receiving user id, trustee verifies the rights of the requested user with the creator access policy *X* and provide token γ. The user receives private key from the nearest KDC for decrypting the encrypted data.The user's credentials were stored in the client itself. During download the file the cloud will authenticate the user whether the user is valid to download the file. But the cloud doesn't have any attributes or the details of the user.

C. Key Distribution Center

In Our project, we can deploy two or more KDC at different places of the world. On registering with a trustee, user will receive attributes and Keys from the nearest KDC. Trustee is assumed to be

honest. KDC generates two separate keys for both encryption and decryption based on user requested id or attributes.

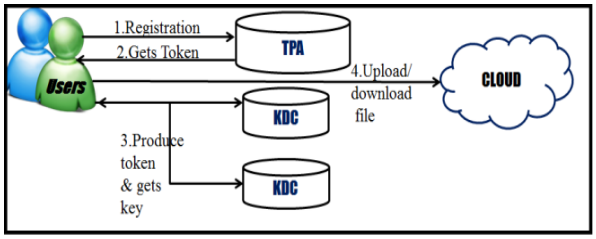


Fig. 3 Decentralized KDC architecture

D. Security Model

Trustee will generate token (γ) only to the registered user which indicates he is a valid user to access the data. Token (γ) contains signature of trustee indicating he is a valid user, id of the user and time stamp. A user can only write provided the cloud is able to validate its access claim. An invalid user cannot receive attributes from a KDC, if it does not have the credentials from the trustee. If a user's credentials are revoked, then it cannot replace data with previous stale data, thus preventing replay attacks.

E. User Revocation

It should be ensured that users must not have the ability to access data, even if they possess matching set of attributes. For this reason, the owners should change the stored data and send updated information to other users. The set of attributes Iu possessed by the revoked user Uu is noted and all users change their stored data that have attributes *i*€ Iu.

In [13], revocation involved changing the public and secret keys of the minimal set of attributes which are required to decrypt the data. We do not consider this approach because here different data are encrypted by the same set of attributes, so such a minimal set of attributes is different for different users. Therefore, this does not apply to our model. Once the attributes Iu are identified, all data that possess the attributes are collected. For each such data record, the following steps are then carried out:
1. A new value of s, Snew € Zq is selected.
2. The first entry of vector Vnew is changed to Snew.
3. λx= RxVnew is calculated, for each row x corresponding to leaf attributes in Iu.
4. C1,x is recalculated for x.
5. New value of C1,x is securely transmitted to the cloud.
6. New C0 =Me(g, g)Snew is calculated and stored in the cloud.
7. New value of C1,x is not stored with the data, but is transmitted to users, who wish to decrypt.
We note here that the new value of C1,x is not stored in the cloud but transmitted to the non-revoked users who have attribute corresponding to x. This prevents a revoked user to decrypt the new value of C0 and get back the message.

V. DAC CONTRIBUTIONS

The main contributions of this paper are the following:
1. Distributed access control of stored data so that only authorized users with valid attributes can access.
2. Authentication of users who store and modify their data on the cloud.

3. The architecture is decentralized, meaning that there can be several KDCs for key management.
4. The access control and authentication are both collusion resistant,
5. Revoked users cannot access data after they have been revoked.
6. The proposed scheme is resilient to replay attacks. A writer whose attributes and keys have been revoked cannot write back stale information.
7. The protocol supports multiple read and write on the data stored in the cloud.
8. The costs are comparable to the existing centralized approaches.

VI. COMPUTATION COMPLEXITY

In this section, we present the computation complexity of the privacy preserving access control protocol. We will calculate the computations required by users by the cloud. The creator needs to encrypt the message and sign it. Creator needs to calculate one pairing e(g,g). Encryption takes two exponentiations to calculate each of C1,x. So this requires 2mET time, where m is the number of attributes. User needs to calculate three exponentiation to calculate C2,x and C3,x. So time taken for encryption is (3m + 1)E0 + 2mET + TP.

The cloud needs to verify the signature. Time taken to verify is (l +2t) Tþ +l(E1 + E2)+TH. To read, a user needs only to decrypt the ciphertext. This requires 2m pairings to calculate e(H(u), C3,x) and e(skπ(x),u, C2,x) and O(mh) to find the vector c. Decryption takes 2mTþ+ TH + O(mh).

VIII. PERFORMANCE ANALYSIS

A. Time Performance

The performance of this paper was analysed under various file sizes. At first the time performance of this paper is evolved for different file sizes. Then the cryptographic operation time is evolved. The only achievement of this paper is it supports random time duration for any size of files to download.

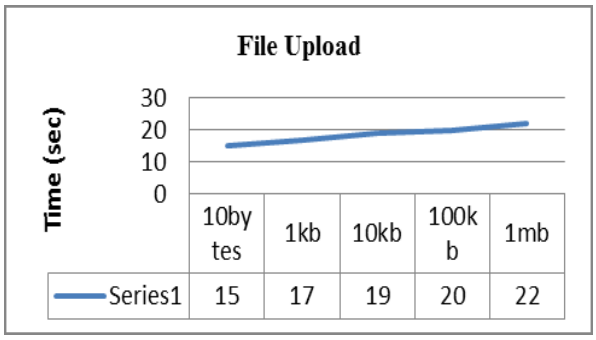
FileSize	Upload (sec)	Download (sec)
10bytes	15	0
1kb	17	3
10kb	19	0
100kb	20	7
1mb	22	7

Table 1.Time performance for transaction on cloud

Fig4:Performance Analysis of the File Upload Process

B. Upload

File uploading time is not a constant one. For same size file the time taking for uploading is randomly different. Using the time taken to upload the file one can identify the encryption standard. To confuse the hacker the random time delay is achieved.



C. Download

File downloading time is also not a constant one. For same size file the time taking for downloading is randomly different. Using the time taken to download the file one can identify the encryption standard. To confuse the hacker the random time delay is achieved.

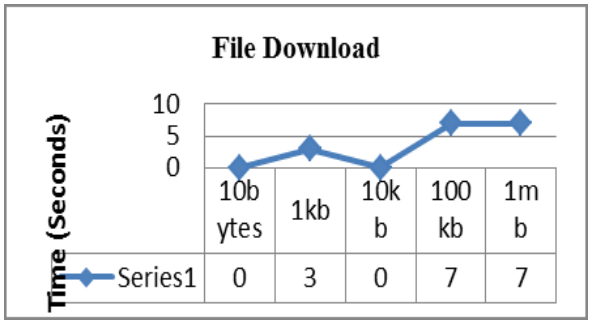


Fig5:PerformancAnalysis of File Download Process

VII. COMPARISON WITH OTHER ACCESS CONTROL SCHEMES IN CLOUD

We compare our scheme with other access control schemes (in Table 1) and show that our scheme supports many features that the other schemes did not support. 1-W-M-R means that only one user can write while many users can read.M-W-M R means that many users can write and read. We see that most schemes do not support many writes which is supported by our scheme. Our scheme is robust and decentralized; most of the others are centralized. Our scheme also supports privacy preserving authentication, which is not supported by others. Most of the schemes do not support user revocation, which our scheme does.

TABLE 2
Comparison of Our Scheme with Existing Access Control Schemes

Schemes	Fine-grained access control	Centralized/ Decentralized	Write/read access	Type of access control	Privacy preserving authentication	User revocation?
[38]	Yes	Centralized	1-W-M-R	Symmetric key cryptography	No authentication	No
[12]	Yes	Centralized	1-W-M-R	ABE	No authentication	No
[13]	Yes	Centralized	1-W-M-R	ABE	No authentication	No
[16]	Yes	Decentralized	1-W-M-R	ABE	No authentication	Yes
[33]	Yes	Centralized	1-W-M-R	ABE	No authentication	No
[34]	Yes	Decentralized	1-W-M-R	ABE	Not privacy preserving	Yes
[15]	Yes	Centralized	M-W-M-R	ABE	Authentication	No
Ours	Yes	Decentralized	M-W-M-R	ABE	Authentication	Yes

IX. CONCLUSION

We have presented a decentralized access control technique with anonymous authentication, which provides user revocation and prevents replay attacks. The cloud does not know the identity of the user who stores information, but only verifies the user's credentials. Key distribution is done in a decentralized way. One limitation is that the cloud knows the access policy for each record stored in the cloud.

X. REFERENCES

[1] M. Li, S. Yu, K. Ren, and W. Lou, "Securing Personal Health Records in Cloud Computing: Patient-Centric and Fine-Grained Data Access Control in Multi-Owner Settings," Proc. Sixth Int'l ICST Conf. Security and Privacy in Comm. Networks (SecureComm),

[2] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Toward Secure and Dependable Storage Services in Cloud Computing," IEEE Trans. Services Computing, vol. 5, no. 2, Apr.- June 2012.

[3] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy Keyword Search Over Encrypted Data in Cloud Computing," Proc. IEEE INFOCOM, pp. 441-445, 2010.

[4] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," Proc. 14th Int'l Conf. Financial Cryptography and Data Security, pp. 136- 149, 2010. [5] H. Li, Y. Dai, L. Tian, and H. Yang, "Identity-Based Authentication for Cloud Computing," Proc. First Int'l Conf. Cloud Computing (CloudCom), pp. 157-166, 2009.

[6] C. Gentry, "A Fully Homomorphic Encryption Scheme," PhD dissertation, Stanford Univ., http://www.crypto.stanford.edu/craig, 2009.

[7] A.-R. Sadeghi, T. Schneider, and M. Winandy, "Token-Based Cloud Computing," Proc. Third Int'l Conf. Trust and Trustworthy Computing (TRUST), pp. 417-429, 2010.

[8] D.F. Ferraiolo and D.R. Kuhn, "Role-Based Access Controls," Proc. 15th Nat'l Computer Security Conf., 1992.

[9] http://securesoftwaredev.com/2012/08/20/xacml-in-the-cloud, 2013.

[10] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-Based Access Control in Social Networks with Efficient Revocation," Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS), 2011.

[11] X. Boyen, "Mesh Signatures," Proc. 26th Ann. Int'l Conf. Advances in Cryptology (EUROCRYPT), pp. 210-227, 2007.

[12] D. Chaum and E.V. Heyst, "Group Signatures," Proc. Ann. Int'l Conf. Advances in Cryptology (EUROCRYPT), pp. 257-265, 1991.

[13] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute Based Data Sharing with Attribute Revocation," ACM Information, Computer and Comm. Security (ASIACCS), pp261, 2010.