ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Enforcing Access Policies Using KP-ABE for Data on Entrusted Cloud

Nehal Pawar

Software Engineer, Dept. of Computer, Pune University, India

ABSTRACT: The mechanism for controlling access to data on Cloud is facing challenges, these challenges would raise great concerns from data owner when they store sensitive information on cloud servers which are not within the same trusted domain as data owners. In old cryptographic method we disclose data decryption keys only to authorized users these solutions inevitably introduce a heavy computation overhead and achieving system scalability and fine-grained data access control, efficient key/user management, user accountability was also an issue. This paper addresses these challenges by defining and enforcing access policies based on data attributes, and allowing the data Owner to delegate most of the computation tasks involved in fine-grained data access control to third party cloud servers without disclosing the underlying data contents. We achieve this goal by Exploiting and uniquely combining techniques of attribute-based Encryption (ABE), proxy re-encryption, and lazy re-encryption.

KEYWORDS: Access control policy, Attribute based, Fine grained data access, Encryption, Key Policy.

I. INTRODUCTION

Cloud Computing [16]is a promising next-generation IT architecture which provides elastic and unlimited resources, including storage, as services to cloud users. In Cloud Computing users and service providers are almost certain to be from different trust domains. Concerns originate from the fact that cloud servers are usually operated by commercial providers which are likely to be outside of the trusted domain. Data confidential on cloud servers is hence frequently desired when users outsource data for storage in the cloud. Data owner publish data on cloud servers for sharing need fine-grained data access control which user (data consumer) has the access privilege.

Commercial service providers are certain to be outside the trust domain, and are not allowed to learn users sensitive information stored on their servers. To enforce access control it turns out that users cannot rely on commercial data servers policies like traditional access control in which reference monitors should be fully trusted. This paper addresses the issue of securing data sharing on entrusted storage by novel public-key cryptography – Attribute-Based Encryption (ABE) methods to help users enforce data access policies on entrusted cloud domain. The owner of data would like to keep her access policy information secure to servers and users may not want to disclosing their access privilege information to servers. Our proposed schemes hide the data owner's access policy not only to the entrusted servers but also to all the users through study of privacy preservation in ABE.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 reviews proposed system some technique preliminaries pertaining to our construction. Section 4 we analyze our proposed scheme in terms of its security and performance. We conclude the paper in Section 5.

II. RELATED WORK

In [2] Attribute-Based Encryption (ABE) was first proposed by [9] with the name of Fuzzy Identity-Based Encryption, with the original goal of providing an error-tolerant identity-based encryption [10] scheme that uses biometric identities.Some similar implementation is seen in area of "shared cryptographic file systems" and "access control of outsourced data".

In [4], proposed Plutus as a cryptographic file system to secure file storage on untrusted servers. As the complexity of key management is proportional to the total number of file-groups, Plutus is not suitable for the case of fine-grained

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

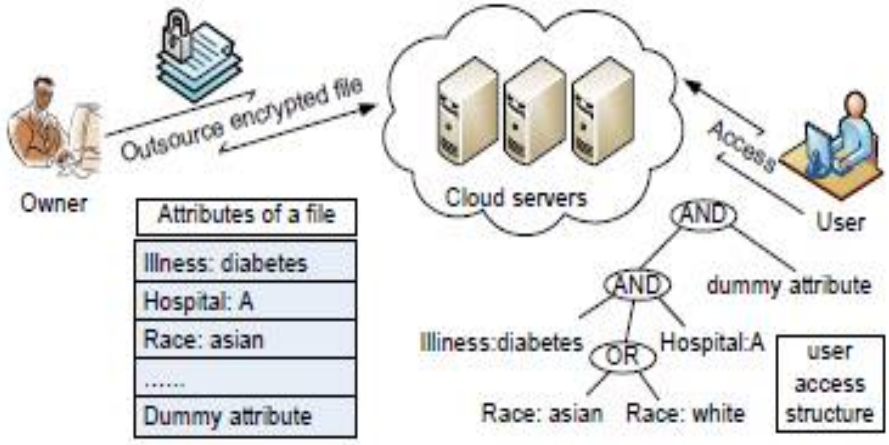
access control in which the number of possible "file-groups" could be huge.

In [5], proposed SiRiUS which is layered over existing file systems such as NFS provides end-to-end security. For this reason of access control, SiRiUS attaches each file with a meta data file that contains the file's access control list (ACL), each entry in this is the encryption of the file's file encryption key (FEK) using the authorized users public key. [7] proposed a secure distributed storage scheme based on proxy re-encryption. The issue with this is that collusion between a malicious server and any single malicious user could expose decryption keys of all the encrypted data and compromise data security of the system completely.In [3], proposed a solution for securing data storage on untrusted servers based on key derivation methods. User access privilege accountability is also not supported.

III. PROPOSED ALGORITHM

As [9] First introduced the public-key cryptography attribute based encryption (ABE) for cryptographically enforced access control. In ABE both the user secret key and the ciphertext are associated with a set of attributes. A user can decrypt the ciphertext if and only if at least a threshold number of attributes overlap between the ciphertext and user secret key.

To enable more general access control, [8] proposed a key-policy attribute-based encryption (KP-ABE) scheme – a variant of ABE. The idea of a KP-ABE scheme is as follows: the ciphertext is associated with a set of attributes and user secret key is embedded along with an access structure which can be any monotonic tree-access structure. A user is able to decrypt a ciphertext if and only if the ciphertext attributes pass the access structure embedded in her secret key. In [11], proposed an enhanced KP-ABE scheme which supports non-monotone access structures reason for not chosing CP-ABE.



In ABE, including KP-ABE and CP-ABE, the authority runs the algorithm *Setup* and *Key Generation* to generate system *MK*, *PK*, and user secret keys. In this paper, we just consider the case of one-writer-and-multiple-reader in untrusted storage for brevity. The only writer is the data owner, who also acts as the authority and is in charge of key generation. This means that the data owner takes the role of both the authority and the encryptor.

Scheme proposed in[1] is composed of seven algorithms: *Setup*, *Enc*, *KeyGen*, *ReKeyGen*, *ReEnc*, *ReKey*, and *Dec*. *Setup*, *KeyGen*, and *ReKeyGen* which are done by the authority while *ReEnc* and *ReKey* are executed by proxy servers. *Enc* and *Dec* are called by encryptors and decryptors respectively. *ReKeyGen* is defined for the authority to generate proxy re-key's.

Setup (1^, n)**.** The setup algorithm is a randomized algorithm. It takes as input the security parameter 1^ *and n*, the length of a user identity. It outputs a master key *MK* and public parameters *PK*.

KeyGen (*S*, *MK*, *PK*). The key generation algorithm is a randomized algorithm. It takes as input an set of attribute S, the master secret key *MK*, and the public parameters *PK*. It outputs a user secret key *SK*.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

A.User revocation in KP-ABE application

User secret keys should be updated accordingly for data access. New data files will be encrypted with the new public key, and existing data files should be re-encrypted to prevent revoked users from decrypting using their old version keys.On each user/attribute revocation event, the authority redefines the corresponding keys for the attributes and generates proxy re-key's for the updated master key components, with which the proxy servers are able to securely update user secret keys to the latest version on behalf of the data authority without obtaining the users' decryption capabilities.

ReKeyGen(y, *MK*) It takes as input an attribute set *y* that includes attributes for update, and current master key *MK*. It outputs the new master key *MK'*, the new public key *PK'* (computation of *PK'* can be delegated to proxy servers), and a set of proxy re-key's *rk* for all the attributes in the attribute universe *U*. *version* is increased by 1. Note that, for attributes in set *U-y*, their proxy re-key's are set as 1 in *rk*.*ReEnc* and *ReKey* will be used by the proxy servers to re-encrypt data files and update user secret keys respectively. In our scheme we also define version information *ver* indicating the evolution of the system master key as stated: initially it is set to one; whenever an attribute revocation event occurs and the system master key is redefined, it increases by one. The system public key, ciphertexts, user secret keys, and proxy re-key's are all tagged with the version information indicating which version of system master key they comply with.

ReEnc(*CT*,*rk*,*b*) It takes as input a ciphertext *CT*, the set of proxy re-key's *rk* having the same version with *CT*, a set of attributes b which includes all the attributes in *CT*'s access structure with proxy re-key not being 1 in *rk*. It outputs a re-encrypted ciphertext *CT'* with the same access structure as *CT*.

ReKey($\bar{}D$; *rk*; μ) It takes as input the component $\bar{}D$ of a user secret key *SK*, the set of proxy re-key's *rk* having the same version with *SK*, and a set of attributes μ which includes all the attributes in *SK* with proxy re-key not being 1 in *rk*. It outputs updated user secret key components $\bar{}D'$.

B. User Accountability

The security goal of our construction is to build such a KP-ABE scheme in which 1) user with incorrect decryption key is not able to tell a single bit of the message, and 2) given a pirate device, the authority is able to trick it into decrypting tracing ciphertexts and thus finding the identity of the owner of the decryption key held by this device.

Enc (*M*, *y*, *PK*). The encryption algorithm is a randomized algorithm. It takes as input a message *M*, a set of attributes *y*, and the public parameters *PK*. It outputs a ciphertext *E*. On different input *y*, this algorithm can be used either for normal (non-tracing) operations of content distribution, or for the purpose of tracing.

Dec (*E*, *SK*, *PK*). The decryption algorithm is a deterministic algorithm. It takes as input the ciphertext *E* for a set of attributes *y*, a user secret key *SK* for an access structure *T*, and the public parameters *PK*. If *y*= *T*, i.e., *y* satisfies *T*, it outputs the message *M*m

C. Definition of Attributes and Access structure

We define three set of attributes: *public normal attributes*, *hidden normal attributes* and *hidden identity-related attributes*. We denote the universe of each of them by *UPN; UHN*, and *UHID* respectively. The letter *P* in the subscription denotes the word "public", *H* stands for "hidden", *N* means "normal", and *ID* is the shortform of "identity". *UPN* and *UHN* contain attributes to be used by normal encryptions. *UHID* contains identity-related attributes for detaling the suspected user's identity and is used for tracing. In ciphertexts, the associated attributes from *UHN* and *UHID* have to be conceled such that any receiver is not able to tell which and how many of them are used, while attributes from *UPN* are public. Each attribute in *UHID* has two occurrences, one for bit value 0 and the other for bit value 1. We denote the set of these three types of attributes in a ciphertext by *PN*, *HN* and *HID* interested by the encryptor. We differentiate two kinds of attributes: *application level attributes* and *algorithm level attributes*. Application level attributes indicate to those meaningful to human being, e.g., occupation, skill, rank etc. Algorithm level attributes refer to the ones suitable for computer to translate. Application level attributes can be mapped to algorithm level attributes. In this work, an attribute refers to an algorithm level attribute which is clarified in a way that it has two possible outcomes: *positive* and *negative* which are denoted with symbols *Atti;*1 and *Atti;*0.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

D. ACCESS STRUCTURE

Our definition of the access structure (implemented using an access tree) is the same as KP-ABE [8], i.e., each interior tree's node is a threshold gate and the leaves are associated with attributes. However, our structure has the following restrictions on the access structure: (1) all access structure should deal with all the conceled attributes and all of them should appear on the second layer of the tree; (2) the root node has to be an AND gate; (3) all the attributes from *UPN* should be seen in a subtree which is denoted by *TR*. Nodes impilict to the subtree *TR* could be any kind of threshold gates. In addition, each non-root the form of access structure node has a unique index assigened by its parent. For the convenience of representation, we will denote a node *x*'s parent by *xpa* and *x*'s index by *idx*(*x*).

Access tree *T*. Let *T* be a tree representing an access structure. Tree's each non-leaf node represents a threshold gate, described by its children and a threshold value. When *numx* is number of children of a node *x* and *kx* is the threshold value, then $0 < kx \cdot numx$. If $kx = 1$, then threshold gate is an OR gate and when $kx = numx$, it is an AND gate. Each leaf node *x* of the tree is detailed by an attribute and a threshold value $kx = 1$. To support 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 determined only if *x* is a leaf node and represents the attribute associated with the leaf node *x* in the tree. The access tree *T* also defines a sequence between the children of every node, i.e; the children of a node are numbered from 1 to *num*. The function index(*x*) returns such a number releted with the node *x*. Where nodes are uniquely assigned the index values in the access structure for a given key in an random manner. Complying with an access tree. Let *T* be an access tree with root *r*. The subtree denote by *Tx* of *T* rooted at the node *x*. Hence *T* is the same as *Tr*. If a set of attributes *y* satisfies the access tree *Tx*, we denote it as *Tx*(y) = 1. We compute *Tx*(*y*) recursively as follows. If *x* is a non-leaf node, evaluate *Tx'*(*y*) for all children *x'* of node *x*. *Tx*(*y*) returns 1 if and only if at least *kx* children return 1. If *x* is a leaf node, then *Tx*(*y*) returns 1 if and only if att(*x*) *E y*:

IV. RESULT ANALYSIS

In KP-ABE, ciphertexts are associated with attributes, while us secret keys are defined with access structures on attributes. If only theciphertext attributes satisfy a user's access structure, can he decrypt. KP-ABE is suitable for applications such as IT company sharing data over server , in which user access privileges are defined over content attributes and could be based on their designation, project they work on and department. In these application scenarios, the issue of key revocation also exists in which a user currently is allowed to access any project with name "PPR", "NCL", or "DTE"' provided by Department 'Testing'. The system administrator now wants to disable the user's access privilege on project with name "NCL" for unknown reason. For this intention, it is required to cancel the corresponding component of the user's secret key.

The basic construction of current KP-ABE scheme [8] also defines a system master key component *ti* for each attribute *i*. The corresponding public key component is defined as *Ti* = *gti* . Encrypting a message with attribute *i* means including a component *Ts i* into the ciphertext, where *s* is a random number for this ciphertext. In user secret key, the component for attribute *i* has the form of polynomial uniquely defined for the user. Therefore, a secret key component can be revoked in the same way as we did for CP-ABE, i.e., the authority redefines the master key component as *t' i* and give *t'i ti* to proxy servers as the proxy re-key.

V. EFFICIENCY ANALYSIS

In AFKP-ABE, both the ciphertext size and the secret key size are linear to *n*, where *n* is the number of bits in the identity space. As the maximum number of users it can show is $N = 2n$, the compoundness can be written as $O(logN)$, where *N* is the total number of users. To trace apirate, AFKP-ABE needs to try with every user's identity in the system list. With large number of users, the tracing algorithm would be inefficient. To resolve this issue, we can first test withsome normal ciphertexts using combinations of normal attributes. For example, different combinations of attributes like location, age, etc can be used. In practice, this process will hopefully rule out a significant portion of users. Our tracing algorithm can just test over the remaining set of users.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

VI. CONCLUSION AND FUTURE WORK

In this paper, we addressed an important issue of secure data sharing on untrusted storage.A novel PKC Attribute-Based Encryption (ABE) is used to provide cryptographically enforced data access control. With ABE, we are able to enjoy fine-grained access control. We proposed three security enhancing solutions for ABE: The first enhancement we made is to provide efficient user revocation in ABE. With this assumption in which semi-trustable proxy servers are available we uniquely combined the proxy re-encryption technique with ABE and enabled the authority to delegate most laborious tasks to proxy servers. In our second enhancement to ABE, we addressed key abuse attacks and proposed an abuse free KP-ABE (AFKP-ABE) scheme. The complexity of AFKP-ABE in terms of ciphertext size and user secret keys size is just $O(logN)$, where N is the total number of users. Our third enhancement is to provide better privacy preservation for ABE in terms of access policy information protection. With ABE and our enhancement schemes, we presented our solutions for secure data sharing for Cloud Computing. In doing so, we combined our enhanced ABE schemes with techniques such as dummy attribute and lazy re-encryption, and made it possible for both the data owner and users to delegate most computation-intensive operations to powerful cloud servers.

We identify three directions for future work for secure data sharing on untrusted storage as follows. Decentralized Access Control In this paper, there is one cryptosystem in each data application and the data owner acts as the only authority in every cryptosystem. Operation on Encrypted Data When encryption provides data confidentiality, it also greatly limits the flexibility of data operation. Another interesting future work would be taking into account information theoretic techniques from the areas such as database privacy. In order for doing so, one interesting future work would be integrating techniques from trusted computing [13] into the data access control mechanism

REFERENCES

[1] Shucheng Yu on Data Sharing on Untrusted Storage with Attribute-Based Encryption 2010.
[2] R. H. Deng, J. Weng, S. Liu, and K. Chen. Chosen-Ciphertext Secure Proxy Re-encryption without Pairings. In *Proc. of CANS'08*, Berlin, Heidelberg, 2008.
[3] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Over-encryption: Management of Access Control Evolution on Outsourced Data. In *Proc. of VLDB'07*, Vienna, Austria, 2007.
[4] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu. Plutus: Scalable Secure File Sharing on Untrusted Storage. In *Proc. of FAST'03*, Berkeley,California, USA, 2003.
[5] E. Goh, H. Shacham, N. Modadugu, and D. Boneh, "Sirius: Securing remote untrusted storage," in *Proc. of NDSS'03*, 2003.
[6] S. Yu, K. Ren, W. Lou, and J. Li. Defending Against Key Abuse Attacks in KP-ABE Enabled Broadcast Systems. In *Proc. of Securecomm'09*, Athens, Greece, 2009.
[7] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy reencryption schemes with applications to secure distributed storage," in *Proc. of NDSS'05*, 2005.
[8] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. Of CCS'06*, 2006.
[9] A. Sahai and B. Waters. Fuzzy Identity-Based Encryption. In *Proc. of EUROCRYPT' 05*, Aarhus, Denmark, 2005.
[10] D. Boneh and M. Franklin. Identity-Based Encryption from The Weil Pairing. In *Proc. of CRYPTO'01*, Santa Barbara, California, USA, 2001.
[11] R. Ostrovsky, A. Sahai, and B. Waters. "Attribute-based encryption with non-monotonic access structures". In *Proc. of CCS'06*, New York, NY, 2007.
[12] A. Lewko and B. Waters, "Decentralizing Attribute-Based Encryption",http://eprint.iacr.org/2010/351.
[13] Trusted Computing Group http://www.trustedcomputinggroup.org/
[14] Vipul Goyal¤ Omkant Pandeyy Amit Sahaiz Brent Waters on Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data.
http://computer.howstuffworks.com/cloud-computing/cloud-computing.htm

BIOGRAPHY

Nehal Pawar received the B.E. degree in Computer Engineering from All India Shree Shivaji Memorial Socity Institute of Information Technology in 2014. He is now with Datamatics Global Services Ltd.