IJournals: International Journal of Software & Hardware Research in Engineering
ISSN-2347-4890
Volume 3 Issue 9 September, 2015

Decentralized access control with policy hiding to store data in clouds.

Anusha R[1]; Dr.R.ChinaAppala Naidu[2]

Mtech Student [1], Professor[2]

Dept of Computer Science and Engineering[1,2]

St.Martin's Engineering College, Dhulapally, Kompally, Hyderabad,India[1,2].

amanuknr@gmail.com[1];chanr789@gmail.com[2]

ABSTRACT

Cloud computing enables the users to use the IT capabilities like servers, network, storage, database servers etc.) in an on demand basis, pay per use model. Of all security requirements in cloud computing, access control is important to avoid unauthorized access to systems and protect organizations properties. So we discuss a scheme with decentralized access control with anonymous authentication and hides the attributes and access policy to store data securely in clouds. This scheme checks the authenticity of the user without using his identity to store data. This allows only legitimate users to decrypt the information stored. This prevents Replay attacks, and allows user revocation. In order to improve the effectiveness of the system, a new Attribute Based Encryption algorithm is proposed which hides the attributes and access policies. This stops other users in extracting information from the cipher. This is achieved by the use of hash functions and polynomial functions to hide the attributes and access policies. Data privacy and policy privacy are also ensured.

Keywords: Cloud computing, Attribute Based Encryption, hash functions, polynomial functions, access policy, Data privacy, policy privacy

1. INTRODUCTION

Cloud computing enables the users to use the IT related capabilities like servers, network, storage, database servers etc. in an on demand basis, pay per use model. It provides services on-demand such as Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS).There are a number of challenges associated with cloud computing. They are data security, malicious insider, cyber-attacks etc. Among all security requirements of cloud computing, access control is necessary to avoid unauthorized access of systems and protect organizations properties.

Another challenge in cloud computing is maintaining the secrecy of the data. Cloud storage requires the fine-grained access control. Cloud stores and access information from cloud storage. When used as public service, controlled data sharing is difficult. This can be obtained using encryption and decryption mechanisms. But

traditional public-key and secret key mechanisms can be used with fixed identity of users and not for dynamic users. To avoid this Attribute Based Encryption (ABE) is proposed.

The ABE system is a predicate based encryption. The encryption done with Access Policies which are defined by the Access trees. In threshold-access tree, each non-leaf nodes agrees to a threshold value and the leaf-nodes contains the attributes. If the thresholds of the non-leaf nodes are satisfied, then the leaf nodes attributes can be used to access the data else cannot. This access trees uses the Linear Secret Sharing Scheme (LSSS) matrix to create the attributes for encryption. The Access trees are related to the cipher text. ABE system creates the keys using the attributes set. If these attributes matches with the cipher text, then the decryption can be done else the decryption cannot be done. Attribute based encryption is an efficient and reliable method for sharing data in clouds. But there are chances for the access policies of ABE system to be revealed which leads to unauthorised decryption of cipher text. Thus the proposed scheme hides the access policies and attributes.

1.1 Contribution of paper

In this paper, the ABE system which is secure in the form of data policies and attributes is used for encryption while storing data in clouds. Using this scheme other users can't get any information on access policies used and the attributes used to create the keys as it ensures secrecy of data, attributes and access policies.

2. RELATED WORKS

To overcome the problems in identity based encryption (IBE), the Attribute Based Encryption (ABE) system is introduced. The ABE system implements the fine-grained access control systems. Identity based encryption system creates public key based on some exclusive information about the identity of the user (e.g. a user's email address, mobile number. but these are static. IBE is difficult to be used on dynamic or large scale organization. Thus ABE came up where secret key of a user and the cipher text depends on some attributes (e.g. the country he lives) the decryption of a cipher text is can be done only if attribute set of the user key matches the attributes of the cipher text. But here access policies and attributes are revealed.

Functional encryption (FE) is introduced to hide the access policy and attributes. Functional encryption generates secret keys using access structures. During this process, the access policies are associated with the keys. It enables hiding of Policies and attributes. Data owner should believe the authority who describes the access structures [7]. Cloud mask hides the attributes and policies. Cloud mask act as data managers, storage service, users. Data manager performs encryption using access policy and hides the attributes and access policies. Storage service stores the documents from where user access them. However, it is difficult for data managers maintain dynamic access policies[4].

Predicate based encryption is another technique used to hide the access policies and attributes which is similar to ABE. PBE uses predicates, i.e. set of access rules. If the access rules are satisfied with the keys then the decryption can be performed.

3. CLOUD ARCHITECTURE

Cloud computing provides the on-demand self-services based on the pay of use ie, users should

pay based on the services they used. These services are classified as platform as a service (PaaS), infrastructure as a service (IaaS), and software as a service (SaaS).

3.1 Cloud storage

Cloud storage enables digital data storage in multiple servers and the environment is owned and managed by a hosting company. These cloud storage providers check the data are available and accessible to users with protection. The needy utilize these storage capacity to store their data.

3.2 Security requirements

Data confidentiality ensures that only the authorized users can access the data. Unauthorized users do not satisfy the attributes with access policies. Attribute based encryption must satisfy policy secrecy and attribute secrecy, by hiding the policy and attributes from users thus ensuring full security in the attribute based encryption system

Collision resistance is the inability of the users to decrypt the cipher text by combining attributes in the attribute based encryption. Access policy are defined by combining several attributes. This type of collision should be avoided from the attribute based encryption.

4. PREREQUISITE OF CRYPTOGRAPHY

4.1 Access tree

A tree-access structure has attributes stored in leaves. A user can decrypt a cipher text if the attributes corresponding to that cipher text match the key's access structure [1],[2],[3]. For instance, if P has the key associated with the access structure "a AND b", and Q has the key associated with the access structure "b AND c", we won't allow them to decrypt a cipher text whose only attribute is b by colluding. Access structure is defined by a set of attributes A. Subset of attributes A is authorized attributes and others unauthorized.

Let the access tree be *A*. Each node *u* has *n* number of children. The threshold value of node u is *l* . The threshold of each node is of range $0<l<$ n.

In this Access tree, the parent of a node *y* is given by *Parent(y)*.The attributes of a leaf node *y* is given by function *Attribute(y)*. All the children of access tree *A* are uniquely numbered from 1 to n. The function index(y) gives the number of the node *y*. Thus for a given key, the index values are arbitrarily and uniquely assigned to the nodes of the access tree

Let the access tree be *A*. The subtree at node *y* of access tree is *Ay* If the attribute set *Attr* is satisfied by the node *y* and then *Ay(Attr)=l*. It can be repeated in each subtree of access tree.

Ay(Attr) is 1 iff at least *l* children return 1. If y is a leaf node, then *Ay(Attr)* is 1 if and only if Attribute(y)∈Attr.

4.2 Bilinear Maps

Let G be a cyclic group of prime order q generated by g. Let G_T be a group of order q. We can define the map e: $G \times G \rightarrow G_T$. The map satisfies the following properties:

Let G1 and G2 be two multiplicative cyclic groups of prime order p. Let g be a generator of G1 and e be a bilinear map, e: $G1 \times G1 \rightarrow G2$. The bilinear map e has the following properties:

[1] Bilinearity: for all c,d∈G1 and a,b∈Zp, we have $e(c^a,d^b) = e(c,d)^{ab}$.

[2] Non-degeneracy: e(g,g)!= 1.

4.3 The Bilinear Diffie-Hellman (BDH) algorithm

Let a,b,z∈Zp be chosen at random and g be a generator of G1. The decisional BDH assumption is that no probabilistic polynomial-time algorithm B can distinguish the tuple $(A = g^a, B = g^b, e(g,g)^{ab})$ from the tuple $(A = g^a, B = g^b, e(g,g)^z)$ with more than a negligible advantage.

4.4 ABE algorithm:

Implemented using the following algorithms:

Param(PK, MK) => The key generation centre creates the public key and master keys

Keygen(set of attributes, secret key) =>generates secret key for each user. In these key generation, a set of attributes and secret keys are taken as the inputs and gives secret keys for each user.

Encrypt (public_key, master_public_key, message, Access policies) =>cipher text. The message is encrypted using public key, master key and access policies to create cipher text.

Decrypt (Cipher text, Secret key) =>Message. In this phase cipher text is decrypted using secret keys to give original message.

5. PROPOSED SCHEME

Attribute based encryption uses bilinear mapping defined as G1 x G1 => G2, where G1 and G2 are cyclic groups. Access tree is implemented using secret sharing where a secret value S is shared among all the attributes of the access tree. The shares are determined by a polynomial P example, share of jth attribute is (j, P(j)). Initially the secret key is P(0)=S. Then the polynomial is determined by Lagrange interpolation. The Lagrange polynomial values can be identified by,

$$P(i)=\Sigma\, y_j p_j(i)$$

$$\text{Lagrange coefficient is } p_j(i) = \prod \frac{x-x_i}{x_k-x_i}.$$

5.1 Key generation for KDC.

The Key distribution centre KDC, creates the public key and private key. The KGC selects the group G1, G2 with mapping G1 x G1 => G2, g is the generator of the group; a hash function to map the user identity, $\{0, 1\}^* \rightarrow G1$. Each KDC Aj has a set of attributes Lj which is disjoint from Li of another KDC Ai. KDC choose 2 random numbers q,r.

Public key $P_k = \{G1, g, h=g^r, e(g,g)^q\}$

Secret key $S_k = \{r, g^q\}$

The KDC then publishes its public key.

5.2 User key generation

When a user *u* approach KDC *Ai* for the secret key, it will generate a set of attributes *I[i,u]* for the user and a secret key for each attribute.

The secret key is given by:

$sk_{i,u} = g^{pi} H(u)^{qi,}$

This key is transferred using public key of user so only valid user can decrypt it.

5.3 Encryption

To store user data in clouds, they must be encrypted using access policies. Access policies are hidden by representing them by polynomials at each node of access tree [2].This ensures data hiding and policy hiding. These polynomials at each leaf node are

$Oj = e((g^q)^p, H(Plain\_Attribute))$

For each node in the access tree, a polynomial q_x is chosen in top-down fashion from the root to leaf. For each node x in the tree, the degree d_x of the polynomial q_x must be: $d_x = k_x - 1$, k_x is the threshold value of the node. Now, for the root node r, set $q_r(0) = y$, where y is a randomly selected number, also $d_r$of the polynomial q_r randomly to define it completely. For any other node x, Set $q_x(0) = q_{parent(x)}(index(x))$ and choose d_x randomly to define q_x completely. Once the polynomials are decided, for each leaf node x,the secret value is obtained as follows and given to the user:

$S_x = g^{qx(0)/ti}$ where $t_i$is a random number chosen uniformly.

To encrypt a message M ∈ G2 under a set of attributes *r*, choose a random value s∈Zp and publish the ciphertext as:

$C = (r, E' = Me(g,g)^y, \{E_i = T_i^s\}_{i \in r})$.

5.4 Decryption

In the attribute based encryption, decryption is done in 2 levels. First, the user retrieves the attributes from the access tree. It checks the attributes that are common to him and access matrix. It uses the secret key of users for decryption at nodes of access tree.Using the keys obtained here, data is decrypted. This is the second phase.

Receiver Ur takes ciphertext C, secret keys $\{sk_{i,u}\}$, group G0, and outputs message msg.. It executes the following steps:

[1] Ur calculates the set of attributes common to access matrix and itself.

[2] If there are attributes common with linear combination, then decryption is possible else not.

[3] With these attributes the user decrypts the data.

For a leaf node x Decrypt function takes cipher text, secret key $S_x$and the node x.

$D=(e(S_x,E_i)) = g^{qx(0)/ti} g^{s \cdot ti} = e(g,g)^{s \cdot qx(0)}$ if i∈r else null.

The decryption algorithm after recovering $Me(g,g)^y$ after the satisfaction of access tree recovers the original message M from this.

5.5 Decentralized access control

A user Uu registers in one or more trustees which gives the token *tok*= (u, Kbase, K0, Sign); Sign is the signature on $u||K_{base}$ signed with the TSig - trustees private key. The user shows it to the KDCs to get the attribute and secret keys. KDC Ai calculates key for the attributes x as

$Kx = K^{1/(a+bx)}_{base}$, where(a,b) ∈ secret key of Ai. The user also receives secret keys $sk_{i,u}$ for encrypting messages. The user then creates an access policy X The message is then encrypted under the access policy as

Cipher C= ABE.Encrypt(M,X) The user then generates a claim policy Y for the cloud to

authenticate it.The msg M is then appended with timestamp *ts* to prevent replay attacks. It then signs the message and creates the signature of the message as σ = ABS.Sign(Trustee$_{Publickey}$,KDC$_{Public\ key}$, token, signing key, message, access claim).

The following information is then sent to the cloud. c =(C, *ts*, σ, Y).

The cloud verifies the access claim of received information. If authentication is success the message is stored into cloud else it is discarded.

When a user access data from the cloud, it transfers ciphertext C using secure protocol. The user then decrypts the data using the decryption algorithm ABE.Decrypt(C, {sk$_{i,u}$}) and the message M is calculated[1].

To write to an existing file, the user its claim policy with its message. This is verified by the cloud and allows to write only if user is authentic.

5. CONCLUSION

This scheme ensures data confidentiality by the two levels of encryption- use of access structures hides the access policies as only with matching policies can the user decrypt the data. Collusion resistance is also achieved as random values are chosen to generate the keys and attributes. Thus hiding policies also prevents collusion. The scheme also uses a decentralized approach to store data in clouds with anonymity. It is also resistant to reply attacks as the messages are always stored with the timestamp appended.

6. REFERENCES

[1] SushmitaRuj, Milos Stojmenovic, Amiya Nayak, "Decentralized Access Control with Anonymous Authentication of Data Stored in Clouds", IEEE Transactions On Parallel and Distributed Systems, vol.25, issue No.02, pp: 384-394, Feb.2014

[2] D.Vaduganathan, S.Ramasami, "Attribute Based Encryption With Attribute Hiding In Cloud Storage",International Journal For Trends In Engineering & Technology, Issn: 2349 – 9303, Volume 3, Issue 3, March 2015

[3] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data", in ACM Conference on Computer and Communications Security, pp. 89–98, 2006

[4] Weimin WEI, Chaochao SUN,Daming LIU, "A Survey of Privacy-preservingAccess Control in Cloud Computing,"Journal of Computer Information Systems, pp. 5829-5846,Jul2014

[5] Dan Boneh, Amit Sahai and Brent Waters, "Functional Encryption: Definitions and Challenges", TCC,pp. 253-273, 2011