Access Policy in Cloud Based On Compound Attributes of System Users

[1]Mrs. Aruna N. S, [2]Anu C S

Assistant Professor,
Department of CSE
[1]Global Academy of Technology, [2]Bapuji Institute of Engineering & Technology
Bengaluru, India

Abstract—Cloud computing has emerged as one of the most influential paradigms in the IT industry in recent years. Since this new computing technology requires users to entrust their valuable data to cloud providers, there have been increasing security and privacy concerns on outsourced data. Several schemes employing attribute-based encryption (ABE) have been proposed for access control of outsourced data in cloud computing; however, most of them suffer from inflexibility in implementing complex access control policies. Scalable, flexible and fine-grained access control are important and challenging issues concerning about data in cloud computing. A Hierarchical Attribute-Set-Based Encryption (HASBE) is designed and implemented which extends Ciphertext-Policy Attribute-Set-Based Encryption (CP-ASBE) with a hierarchical structure of users. This scheme achieves scalability due to its hierarchical structure and fine-grained access control of out-sourced data in cloud computing by supporting compound attributes of Attribute-Set Based Encryption (ASBE).

Index Terms—HASBE, CP-ASBE, ASBE, Encryption, ciphertext

I. INTRODUCTION

When a set of photos are stored online instead of on a home computer, or use webmail or a social networking site, a "cloud computing" service is being used. If an organization, wants to use, an online invoicing service instead of updating the in-house one that has been using for many years then such a service is a "cloud computing" service. Instead of keeping data on your own hard drive or updating applications for your needs, a service may be used over the Internet, at another location, to store the information or use its applications. The following definition of cloud computing is given by the U.S. National Institute of Standards and Technology (NIST):

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models [1].

While there are benefits, there are privacy and security concerns too. Data is travelling over the Internet and is stored in remote locations. In addition, cloud providers often serve multiple customers simultaneously. All of this may raise the scale of exposure to possible breaches, both accidental and deliberate. Concerns have been raised by many that cloud computing may lead to "function creep" — uses of data by cloud providers that were not anticipated when the information was originally collected and for which consent has typically not been obtained. Given how inexpensive it is to keep data, there is little incentive to remove the information from the cloud and more reasons to find other things to do with it.

Security issues, the need to segregate data when dealing with providers that serve multiple customers, potential secondary uses of the data—these are areas that organizations should keep in mind when considering a cloud provider and when negotiating contracts or reviewing terms of service with a cloud provider. Given that the organization transferring this information to the provider is ultimately accountable for its protection, it needs to ensure that the personal information is appropriate handled.

II. LITERATURE SURVEY

There is a trend for sensitive user data to be stored by third parties on the Internet. With security concerns hovering over adoption of cloud computing, many access control models have been proposed.Bell-La Padula (BLP) [3] and BiBa [4] are two famous security models which proposes the two fundamental types of "access" that can occur between subjects and objects are termed

observation and alteration. Bell and La Padula define the first as the "extraction of information" from an object, and the second as the "insertion of information" into an object. These schemes are only applicable to systems in which data owners and the service providers are within the same trusted domain.

The notion of ABE was first introduced by Sahai and Waters [5] as a new method for fuzzy identity-based encryption. Identity-Based Encryption (IBE) allows for a sender to encrypt a message to an identity without access to a public key certificate. The ability to do public key encryption without certificates has many practical applications.

Yu et al. [6] proposed a Key-Policy Attribute Based Encryption (KP-ABE) which is a public key cryptography primitive for one-to-many communications. J. Bethencourt [7] proposed Ciphertext-Policy Attribute-Based Encryption (CP-ABE) in which a user's private key will be associated with an arbitrary number of attributes expressed as strings.

III. BACKGROUND AND MOTIVATION

Bobba [8] proposed a new form of CP-ABE called, Cipher-text Policy Attribute-Set Based Encryption (CP-ASBE). CP-ABE schemes can only support user attributes that are organized logically as a single set; i.e., users can use all possible combinations of attributes issued in their keys to satisfy policies. CP-ASBE addresses the limitations of CP-ABE by introducing a recursive set based structure on attributes associated with user keys. Specifically CP-ASBE allows, User attributes to be organized into a recursive family of sets and policies that can selectively restrict decrypting users to use attributes from within a single set or allow them to combine attributes from multiple sets.

Thus, by grouping user attributes into sets such that those belonging to a single set have no restrictions on how they can be combined, CP-ASBE can support compound attributes without sacrificing the flexibility to easily specify policies involving the underlying singleton attributes. Similarly, multiple numerical assignments for a given attribute can be supported by placing each assignment in a separate set.

CP-ABE schemes that represent user attributes as a monolithic set in keys whereas CP-ASBE organizes user attributes into a recursive set based structure and allows users to impose dynamic constraints on how those attributes may be combined to satisfy a policy.

Consider attributes for students derived from courses they have taken. Each student has a set of attributes (Course, Year, Grade) for each course she has taken. In the following, consider a simple policy "Students who took a 300 _ Course < 400 in Year _ 2007 and got Grade > 2." Using a CP-ABE scheme for this is challenging because, for instance, a student can take multiple courses and obtain different grades in them. The policy circuit will have to ensure that she cannot mix together attributes from different sets to circumvent the policy.

For each course that the student has taken, let there be a single designed (boolean) attribute that she gets (e.g. cyg:373 2008 4). But the designed policy will have to (unrealistically) anticipate all such attributes that will satisfy the policy (e.g.,cyg:300 2007 3 or cyg:301 2007 3 or . . . or cyg:399 2010 4).

Wang et al. [2] proposed hierarchical attribute-based encryption (HABE) to achieve fine-grained access control in cloud storage services by combining hierarchical identity-based encryption (HIBE) and CP-ABE. This scheme also supports fine-grained access control and fully delegating computation to the cloud providers.

IV. EXISTING SYSTEM

The notion of Attribute Based Encryption (ABE) was first introduced by Sahai and Waters [5] as a new method for fuzzy identity-based encryption. The primary drawback of the scheme in [5] is that its threshold semantics lacks expressibility. Several efforts followed in the literature to try to solve the expressibility problem.

In the ABE scheme, ciphertexts are not encrypted to one particular user as in traditional public key cryptography. Rather, both ciphertexts and users' decryption keys are associated with a set of attributes or a policy over attributes. A user is able to decrypt a ciphertext only if there is a match between his decryption key and the ciphertext. ABE schemes are classified into key-policy attribute- based encryption (KP-ABE) and ciphertext-policy attribute- based encryption (CP-ABE), depending how attributes and policy are associated with ciphertexts and users' decryption keys.

V. DRAWBACKS OF EXISTING SYSTEM

The traditional method to protect sensitive data outsourced to third parties is to store encrypted data on servers, while the decryption keys are disclosed to authorize users only. However, there are several drawbacks about this trivial solution.First of all, such a solution requires an efficient key management mechanism to distribute decryption keys to authorized users, which has been proven to be very difficult. Next, this approach lacks scalability and flexibility; as the number of authorized users becomes large, the solution will not be efficient anymore.

In case a previously legitimate user needs to be revoked, related data has to be re-encrypted and new keys must be distributed to existing legitimate users again. Last but not least, data owners need to be online all the time so as to encrypt or re-encrypt data and distribute keys to authorize users.

VI. PROPOSED SYSTEM

The proposed scheme seamlessly extends the CP- ASBE scheme to handle the hierarchical structure of system users. In this, the system model consists of a system admin, multiple admin, data owners and data consumers. The admin is responsible for generating and distributing keys as well as authorizing the data owner and data consumers.

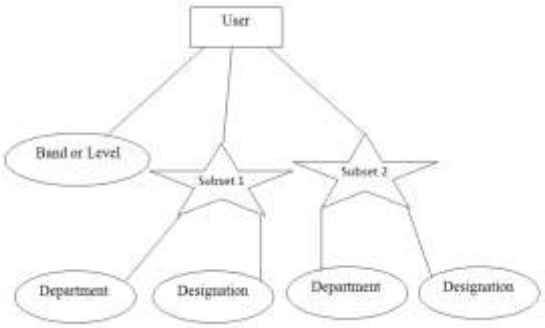

Figure 1 Key Structure based on user attributes set

An admin is responsible for delegating keys to the users in its domain. Each user in the system is assigned a key structure as shown in Figure1 which specifies the attributes associated with the user's decryption key.

The main operations of proposed system are: System Setup, Admin creation (DB), Data owner creation, User creation, File Upload, File Access Structure, File Download and Transactions. As depicted in Figure 2, the cloud computing system under consideration consists of five types of parties: a cloud service provider, data owners, data consumers (users), a number of admin, and system admin. The cloud service provider manages a cloud to provide data storage service. Data owners encrypt their data files and store them in the cloud for sharing with data consumers.

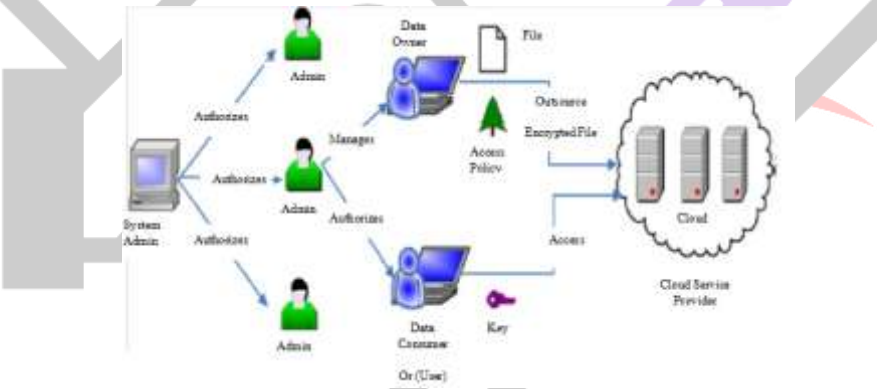

Figure 2 Proposed System Architecture

The cloud is assumed to have abundant storage capacity and computation power. In addition, we assume that data consumers can access data files for reading only.

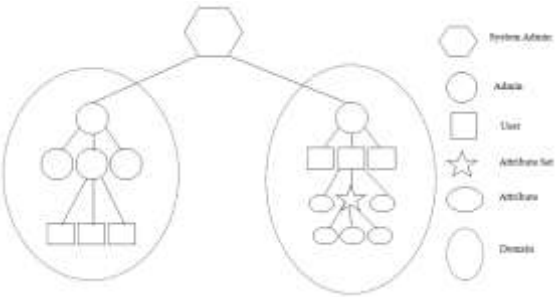

Figure 3 Hierarchical Structure of System Users

We assume that the cloud server provider is untrusted in the sense that it may collude with malicious users (short for data owners/data consumers) to harvest file contents stored in the cloud for its own benefit. In the hierarchical structure of the system users given in Figure 3, each user is associated with a private key, which is kept secretly by the party.

The system admin acts as the root of trust and authorizes the admin. An admin is trusted by its data owner and users that it administrates, but may try to get the private keys of users outside its domain. Users may try to access data files either within or outside the scope of their access privileges, so malicious users may collude with each other to get sensitive files beyond their privileges. So an access policy and private key is based on user attributes. An example access policy structure is shown in the Figure 4.

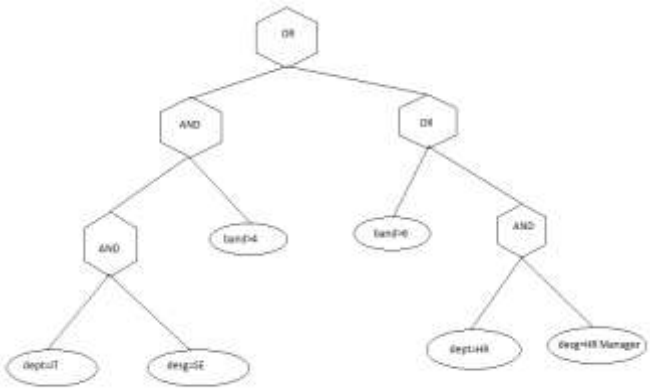

Figure 4 Example Access Policy

VII. IMPLEMENTATION

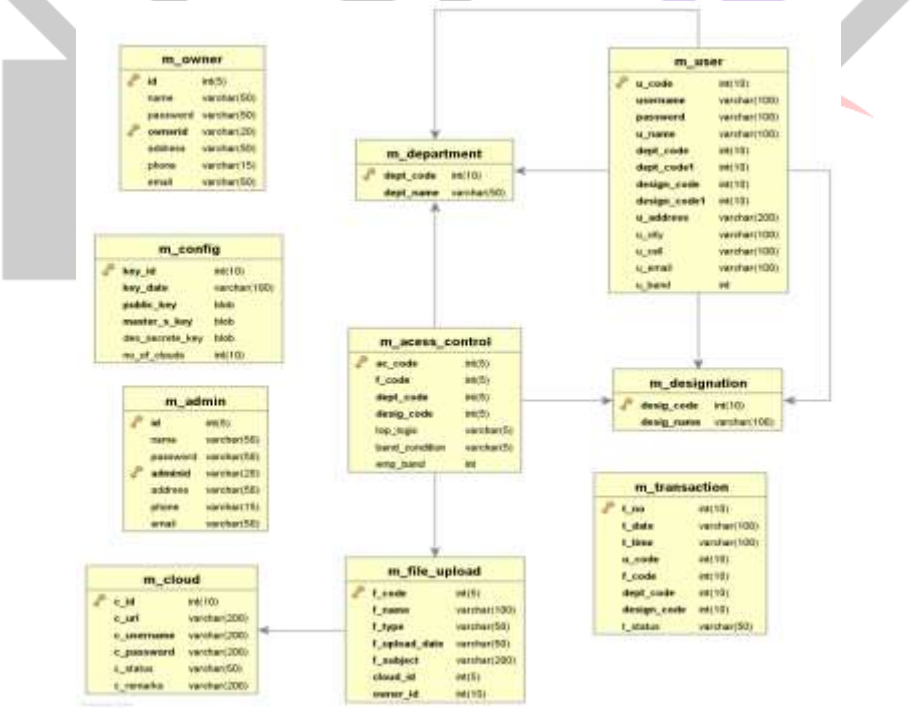

Figure 5 An outline of the proposed system

Preconditions:

1. Create Database schema in MySQL DB Server

2. Application Setup which involves following steps

- Initial data load in DB such as Department information, Designation information, Cloud information, Admin User.
- Tomcat configuration to point Application WEB-INF folder and work directory

Steps: 1. Login into Admin console using Admin user credentials.

2. Create Data owner by providing details of data owner.
3. Create User by providing details of user.
4. Generate Key, which are used for encryption and decryption of File.
5. Logout from Admin console.
6. Login into Data owner console using data owner credentials.
7. Upload data file to cloud by selecting the file from file browser
8. Assign access structure to the file.
9. Logout from Data owner console.
10. Login into user console using user credentials.
11. Download the private key file from the email sent from the System, when user is created or updated.
12. Download the file from cloud by providing private key through file browser.
13. Logout from the User console.

java.security.KeyPairGenerator:class of jre library is used create private key and public key of RSA. Instance of KeyPairGenerator is created by calling getInstance method and passing 'RSA' as string parameter to the method. The KeyPairGenerator instance is initialized by setting 1024 bits. KeyPair is generated by calling generateKeyPair() method of KeyPairGenerator object. The private and public keys are retrieved from KeyPair object.

```
KeyPairGeneratorkpg = KeyPairGenerator.getInstance("RSA");

kpg.initialize(1024);

KeyPair keyPair = kpg.generateKeyPair();

PrivateKey privKey = keyPair.getPrivate();

PublicKey pubKey = keyPair.getPublic();
```

java.security.KeyGeneratorclass of jre library is used create secret key of DES. Instance of KeyGeneratorclass is created by calling getInstance method and passing 'DES' as string parameter to the method. SecretKey is generated by calling generateKey() method of KeyGenerator object.

```
SecretKey key = KeyGenerator.getInstance("DES").generateKey();
```

The generated keys are stored in database through JDBC API.

VIII. SNAPSHOTS OF THE EXPERIMENT

Below are some of the important snapshots of the experiment carried out with above said implementation details. Figure 6 show the setting of access policy done by admin. Figure 7 shows the file access control details. Figure 8 shows a user transaction details. Figure 9 shows a user submitting private key to successfully decrypt and download the file, if private key passes through the access policy set. Figure 10 shows admin can update/ generate the key.

Figure 6 Setting of Access policy

Figure 7 File Access Control Details

Figure 8 A user transaction details

Figure 9 User submitting private key

Figure 10 Generation of key

IX. Conclusion and Future Enhancement

In this paper, we have implemented a system for realizing scalable, flexible, and fine-grained access control of data in cloud computing. This scheme seamlessly incorporates a hierarchical structure of system users by extending CP-ASBE. This not only supports compound attributes due to flexible attribute set combinations, but also achieves efficient user revocation because of multiple value assignments of attributes. It also supports multiple setting of access control on the same file. The decryption keys are generated per user, hence the Key management can be handled efficiently.In this project, user is allowed only to read the contents that are stored in cloud. It may be possible to enhance this project for updating the contents with proper notifications to the higher authorities.

REFERENCES

[1] NIST cloud computing, version 15 http://csrc.nist.gov/groups/SNS/cloudcomputing

[2] G.Wang, Q. Liu, and J.Wu, "Hierachical attibute-based encryption for fine-grained access control in cloud storage services," in Proc. ACM Conf. Computer and Communications Security (ACM CCS), Chicago, IL, 2010.

[3] D. E. Bell and L. J. LaPadula, Secure Computer Systems: Unified Exposition and ultics Interpretation The MITRE Corporation, Tech. Rep., 1976.

[4] K. J. Biba, Integrity Considerations for Secure Computer Sytems The MITRE Corporation, Tech. Rep., 1977.

[5] A. Sahai and B. Waters, "Fuzzy identity based encryption," in Proc.Acvances in Cryptology—Eurocrypt, vol. 3494, LNCS, pp. 457–473, 2005.

[6] S. Yu, C. Wang, K. Ren, and W. Lou, "Achiving secure, scalable, and fine-grained data access control in cloud computing," in Proc. IEEE INFOCOM 2010, pp. 534–542, 2010.

[7] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute based encryption," in Proc. IEEE Symp. Security and Privacy, Oakland, CA, 2007

[8] R. Bobba, H. Khurana, and M. Prabhakaran, "Attribute-sets: A practically motivated enhancement to attribute-based encryption," in Proc. ESORICS, Saint Malo, France, 2009.