ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 2, Issue 4, April 2013

Implementing HASBE Scheme for setting up access controls in out-sourced data Clouds

Darwin v tomy, Dhanalakshmi.S, Dr.S.Karthik

***Abstract*— Cloud computing has emerged as one of the most influential paradigms in the IT industry in current years. Since this new computing technology requires users to entrust their valuable data to cloud providers, there have been growing security and privacy concerns on outsourced data. Numerous schemes employing Attribute-Based Encryption (ABE) have been proposed for access control of outsourced data in cloud computing. In order to get scalable, flexible, & fine-grained access control of outsourced data in cloud computing, in this paper, we propose Hierarchical attribute-set-based encryption (HASBE) by extending cipher text-policy Attribute-set-based Encryption (ASBE) with a hierarchical structure of users. We formally prove the security of HASBE based on security of the cipher text-policy attribute-based encryption (CP-ABE) scheme by Bethencourt et al. and analyse its performance and computational complexity. We apply our scheme and show that it is both efficient and flexible in dealing with access control for outsourced data in cloud computing with comprehensive experiments..**

***Index Terms*— Attribute-based Encryption (ABE), Attribute-Set-based Encryption (ASBE), Hierarchical Attribute-Set-based Encryption (HASBE) and Cipher text Policy Attribute-based Encryption (CP-ABE).**

I. INTRODUCTION

CLOUD computing is a new computing paradigm that is built on virtualization, parallel & distributed computing, utility computing, & service-oriented architecture. In the last several years, The cloud computing has emerged as one of the most influential paradigms in the IT industry, and has attracted widespread attention from both academia and industry. Cloud computing holds the ability of providing computing as the fifth utility. The advancement cloud computing resulted in reduced costs and capital expenditures, increased functioning efficiencies, scalability, flexibility, immediate time to market, and so on. Many types' service-oriented cloud computing models have been suggested, including Infrastructure as a Service (IaaS), Platform
as a Service (PaaS), and Software as a Service (SaaS). Several commercial cloud computing systems have been built at different levels, e.g., Amazon's EC2, Amazon's S3 [3], and IBM's Blue Cloud are IaaS systems, while Google App Engine [5] and Yahoo Pig are representative PaaS systems, and Google's Apps and Salesforce's Customer Relation Management (CRM) System belong to SaaS systems. With these Data cloud systems, The enterprise users no longer need to invest in hardware/software systems or hire IT professionals to maintain these IT systems, thus they help us to save cost on IT infrastructure and human resources; on the other hand, computing services provided by cloud computing are being offered at a relatively low price in a pay-as-you-use mode. For example, Amazon's S3 data storage service with 99.99% durability charges only from $0.06 to $0.15 per gigabyte-month, while the early storage cost starts from $1.00 to $3.50 per gigabyte-month according to Zetta Inc. Although the great benefits brought by cloud computing paradigm are exciting for IT companies, academic scientists, and potential cloud users, security problems in cloud computing had become a serious obstacles which, without being properly addressed, will avoid cloud computing extensive applications and usage in the future. One of the conspicuous security concerns is data security and privacy in cloud computing due to its Internet- based data storage and management. In Data cloud, users have to upload their data to the cloud service provider for storage and business operations, although the cloud service provider is usually a commercial enterprise which cannot be totally trusted. Data represents a tremendously important asset for any organization, and the enterprise users will face a serious consequences if its confidential data is disclosed to their business competitors or the public. Thus, cloud users in the first place will make sure that their data are kept confidential to outsiders, including the cloud providers and their potential competitors. This is the initial data security requirement. Data privacy is not the only security requirement. Flexible and fine-grained access control is also strongly desired in the service-oriented cloud computing model. In a health-care information system on a cloud is required to restrict access of protected medical records to eligible doctors and a customer relation management system running on a cloud may allow access of customer information to high-level executives of the company only. In these cases, the access control of sensitive data is either required by legislation (e.g., HIPAA) or company regulations. Access control is a classic security topic which dates back to the 1960s or early 1970s [9], and various access control models have been proposed since then. Among them, Bell-La Padula (BLP) [10] and BiBa [11] are two famous security models. To attain flexible & fine-grained access control, a number of schemes [12]–[15] have been proposed more recently. Unfortunately, these schemes are only appropriate to systems in which data owners and the service providers are within the same trusted domain. Since data owners and the service providers are usually not in the same trusted domain in cloud computing, a new access control

scheme employing attributed-based encryption [16] is proposed by Yu et al. [17], which adopts the so-called key-policy attribute-based encryption (KP-ABE) to enforce fine-grained access control. However, this scheme falls a short of flexibility in attribute management and lacks scalability in dealing with multiple-levels of attribute authorities. We note that in contrast to KP-ABE, ciphertext-policy ABE (CP-ABE) [18] turns out to be well suited for access control due to its expressiveness in describing access control policies. In this paper, we implement a hierarchical attribute-set-based encryption (HASBE) scheme for access control in cloud computing. HASBE covers the ciphertext-policy attribute- set-based encryption (CP-ASBE, or ASBE for short) scheme by Bobba et al. [19] with a hierarchical structure of system users, so as to achieve scalable, flexible & fine-grained access control. The impact of the paper is multifold. First, we show how HASBE extends the ASBE algorithm with a hierarchical structure to improve scalability and flexibility while at the same time inherits the feature of fine-grained access control of ASBE. Second, we establish how to implement a full-fledged access control system for cloud computing based on HASBE. The scheme provides full support for hierarchical user grant, file creation, file deletion, and user revocation in cloud computing. Third, we formally prove the security of the proposed scheme based on the security of the CP-ABE scheme by Bethencourt et al. [18] and analyze its performance in terms of computational overhead. Lastly, we implement HASBE and conduct comprehensive experiments for performance evaluation, and our experiments demonstrate that HASBE has satisfactory performance. The rest of the paper is organized as follows. Section II provides an overview on related work. Then we present our system model and assumptions in Section III. In Section IV, we describe in detail the construction of HASBE and show how it is used in access control of outsourced data in cloud computing. In this section we prove the security of HASBE and analyze its security by comparing with Yu et al.'s scheme. We analyze computation complexity of HASBE and evaluate.

II. RELATED WORK

In this section, we review the notion of attribute-based encryption (ABE), and provide a brief overview of the ASBE scheme by Bobba *et al.* After that, we examine existing access control schemes based on ABE.

A. Attribute-Based Encryption

The notion of ABE was first introduced by Sahai and Waters [20] as a new method for fuzzy identity-based encryption. The primary drawback of the scheme in [20] 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 categorised 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. In a KP-ABE scheme [16], a ciphertext is associated with a set of attributes and a user's decryption key is associated with a monotonic tree access structure. Only if the attributes associated with the ciphertext satisfy the tree access structure, can the user decrypt the ciphertext. In a CP-ABE scheme [18], the roles of ciphertexts and decryption keys are switched; the ciphertext is encrypted with a tree access policy chosen by an encryptor, while the corresponding decryption key is created with respect to a set of attributes. As long as the set of attributes associated with a decryption key satisfies the tree access policy associated with a given cipher text, the key can be used to decrypt the ciphertext. Since users' decryption keys are related with a set of attributes, CP-ABE is practically closer to traditional access control models such as Role-Based Access Control (RBAC) [18]. Thus, it is more natural to apply CP-ABE, instead of KP-ABE, to enforce access control of encrypted data. On the other hand, basic CP-ABE schemes (e.g., [18]) are far from enough to support access control in modern enterprise environments, which require considerable flexibility and efficiency in specifying policies and managing user attributes [19]. In a CP-ABE scheme, decryption keys only support user attributes that are organized logically as a single set, so users can only use all possible combinations of attributes in a single set issued in their keys to satisfy policies. To solve this problem, Bobba et al. [19] introduced ciphertext-policy attribute-set-based encryption (CP-ASBE or ASBE for short). ASBE is an extended form of CP-ABE which organizes user attributes into a recursive set structure. The following one is an example of a key structure of depth 2, which is the depth of the recursive set structure:

{ Dept : CS, Role : Grad - Student,
{ CourseID :101, Role : TA},
{ CourseID : 525, Role : Grad – Student } }.

The above example represents a key structure assigned to a graduate student in CS department of a university, who is the TA for course 101 and has enrolled in course 525. It can be seen that the similar attribute can be assigned multiple values, e.g., the attribute ―Role‖ is assigned value ―TA‖ and ―Grad-
Student‖ in different sets. This feature renders ASBE more versatile and flexible in supporting many practical scenarios. In this example, the graduate student holding such a private key should not be able to combine the attribute ―Role: TA‖ with CourseID:

ASBE can enforce dynamic constraints on combining attributes to satisfy a policy, which provides more flexibility in access control. In the recursive attribute set which is assigned to a user, attributes from the same set can be combined freely, while attributes from different sets can only be combined with the help of translating items, whose function will be described later. When we consider attributes for students derived from courses they have taken. Each and every student has a set of attributes

(Course , Year, Grade)

For each course he has taken. We want to have a policy. For Students the one who took a course that satisfies
300 ≤ Course < 400 and Year ≥ 2009 and Grade > 3

Enforcing such a policy with CP-ABE is difficult, since a student could have taken multiple courses and obtained different grades in them. The encryptor will have to make sure the student cannot select and combine attributes from different kinds of sets to circumvent the policy. In [19], several possible elucidations with plain CP-ABE are described, but none of them is satisfactory. However, using ASBE, we can solve the problem simply by assigning multiple values to the group of attributes in different sets. For each course the student has taken, he gets a separate set of values for the attributes

(Course, Grade, Year) In this way, ASBE can enforce efficient ciphertext policy encryption for situations where existing ABE schemes are inefficient.

Furthermore, ASBE's capability of assigning multiple values to the same attribute enables it to solve the user revocation problem efficiently, which is difficult in CP-ABE. The revocation problem can be solved easily by assigning different expiration times.

The above desirable feature and the recursive key structure is implemented by four algorithms they are , Setup, KeyGen, Encrypt, and Decrypt:

Setup algorithm (MK, PK) ← Setup (1k): is run by the trusted authority or the security administrator. The setup algorithm takes as input a security parameter k and outputs a master secret key MK and a master public key PK.

Key Generation algorithm (SK) ← Key Gen (MK, ω): is run by the trusted authority, and takes as input a set of attributes ω and MK. The algorithm outputs a user secret key SK associated with the attribute set ω**.**

Encryption algorithm (**CT) ← Encrypt (*m*, PK, P)**: is run by the encryptor. For encryption the input of the algorithm is a message file *m*, a master public key PK and an access control policy P, the output of the algorithm is a ciphertext CT encrypted under the access control policy P.

Decryption algorithm (m) ← Decrypt (CT, SK): is run by the decryptor. The input of the algorithm is a ciphertext CT to be decrypted and a user secret key SK. The output of the algorithm is a message *m*, if the attribute set of the secret key satisfies the access policy P under which the message was encrypted, or an error message if the attribute set of the secret key does not satisfies the access policy P under which the message was encrypted.

These algorithms are essentially similar to those of CP-ABE, except some extensions to support recursive key structure. The public key and the master key of ASBE are extended from CPABE to have components supporting recursive key structure. The master key is extended by adding a new secret exponent β_d for depth. The generated private keys are also different in ASBE and CP-ABE. There are translating components that enable attributes translation between different key sets. The missing part of ASBE is the delegation algorithm, which is used in our proposed scheme to construct the hierarchical structure. We accept the same four algorithms of ASBE, and extend ASBE by proposing a new delegation algorithm.

B. Access Control Solutions for Cloud Computing

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. The data owners need to be online all the time so as to encrypt or re-encrypt data and distribute keys to authorize users.

ABE turns out to be a good technique for realizing scalable, flexible, and fine-grained access control solutions. The use of KP-ABE provides fine-grained access control gracefully. Each file is encrypted with a symmetric data encryption key ( ), which is in turn encrypted by a public key corresponding to a set of attributes in KP-ABE, which is generated according to an access structure. The encrypted data file is stored with the corresponding attributes and the encrypted. If the associated attributes of a file stored in the cloud satisfy the access structure of a user's key, then the user is able to decrypt the encrypted, which is used in turn to decrypt the file.

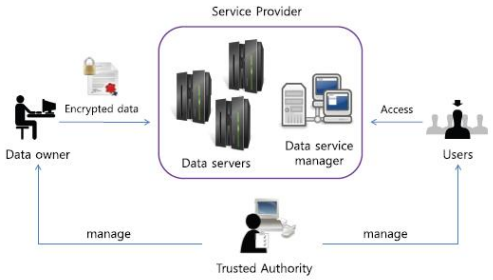

Fig I System Model

The first problem with Yu *et al.*'s scheme is that the encrypt or is not able to decide who can decrypt the encrypted data except choosing descriptive attributes for the data, and has no choice but to trust the key issuer. Furthermore, KP-ABE is not naturally suitable to certain applications. An example of such applications is a type of sophisticated broadcast encryption, where users are described by various attributes and the one whose attributes match a policy associated with a ciphertext can decrypt the ciphertext.

For such an application, a better choice is CP-ABE. Wang *et al.* [21] 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. However, HABE uses disjunctive normal form policy and assumes all attributes in one conjunctive clause are administrated by the same domain master.

New File Creation:

To protect data stored on the cloud, a data owner first encrypts data files and then stores the encrypted data files on the cloud. As in [16], each of the file is encrypted with a symmetric data encryption key that is in turn encrypted with HASBE scheme. Before uploading to the cloud, a data file is handled by the data owner as follows:

- Pick a unique ID for this data file.
- Randomly pick a symmetric data encryption key DEK←K , where is the key space, and encrypt the data.
- Define a tree access structure for the file and encrypt (PK, DEK, T) with using algorithm of HASBE which returns ciphertext. Finally, the encrypted data file is stored on the cloud.

Encrypt (PK, DEK, T). M is the message to encrypt. In the *New File Creation* operation, M is the DEK of a data file. T is the tree access structure. Encrypt algorithm is the same as that of ASBE. The algorithm associates a polynomial with each node in the tree, which is chosen randomly in a top-down manner from the root node. This algorithm computes the Ciphertext as follows:

$$\mathbf{CT} = +\Big(\mathcal{T}, \widetilde{C} = M \cdot e(g,g)^{\alpha \cdot s}, C = h_1^s, \bar{C} = h_2^s, \forall y \in Y : C_y = g^{q_y(0)}, C'_y = H(\mathbf{att}(y))^{q_y(0)}, \forall x \in X : \hat{C}_x = h_2^{q_x(0)}\Big)$$

Where Y denotes the set of leaf nodes in, X denotes the set of translating nodes in the access tree.

User Revocation:

Whenever there is a user to be invalidated, the system must make sure the invalidated user cannot access the associated data files any more. One way to solve this problem is to re-encrypt all the associated data files used to be accessed by the revoked user, but we must also confirm that the other users who still have access privileges to these data files can access them correctly.

HASBE inherits the advantage of ASBE in efficient user revocation. We add an attribute expiration-time X to a user's key, which indicates the time until which the key is considered to be valid. Then the policy linked with data files can include a check on the attribute expiration-time Y as a numerical comparison. The update of user's key and re-encryption of data files can be done as follows:

Key Update. Suppose that there is a user *u*, who is administrated by the domain authority DA*i*. DA*i* preserves some state information about *u*'s key and adds a new value of expiration-time to *u*'s existing key when it wants to update *u*'s key. Then DA*i* computes the secret key components corresponding to the expiration-time attribute and sends them to *u*. Transmission of the secret key components to the user can be accomplished with an out-of-band channel between DA*i* and the user. While DA*i* is required to maintain some state information about user's key, DA*i* avoids the need to create and distribute the entire keys on a frequent basis. This reduces the capacity on DA*i* and saves considerable computing resources.

Data Re-encryption. When the data owner wants to re-encrypt a data file, he changes the value of expiration-time the attribute in the key policy and computes the new ciphertext components C_y and C'_y, where is the leaf node on the access tree corresponding to the expiration-time attribute. Then the data owner sends these new ciphertext components to the cloud and the cloud service

III. Conclusion

In this paper, we introduced the HASBE scheme for realizing scalable, flexible, & fine-grained access control in cloud computing. The HASBE scheme flawlessly incorporates a hierarchical structure of system users by applying a delegation algorithm to ASBE. HASBE not only supports complex attributes due to flexible attribute set combinations, but also achieves effective user revocation because of multiple value assignments of attributes. We formally verified the security of HASBE based on the security of CP-ABE by Bethencourt et al.

Finally, we implemented the proposed scheme, and conducted complete performance analysis and evaluation, which showed its effectiveness and advantages over existing schemes..

.

REFERENCES

[1] R. Buyya, C. ShinYeo, J. Broberg, and I. Brandic, ―Cloud computing and emerging it platforms: Vision,hype, and reality for delivering computing as the 5th utility,‖ 2009.
[2] Amazon Elastic Compute Cloud (Amazon EC2) [Online]. Available: http://aws.amazon.com/ec2/
[3] Amazon Web Services (AWS) [Online]. Available: https://s3.amazonaws. com/
[4] R. Martin, ―IBM brings cloud computing to earth with massive new data centers,‖ *InformationWeek* Aug. 2008.
[5] Google App Engine [Online]. Available: http://code.google.com/appengine/
[6] K. Barlow and J. Lane, ―Like technology from an advanced alien culture: Google apps for education at ASU,‖ 2007.
[7] B. Barbara, ―Salesforce.com: Raising the level of networking,‖ *Inf. Today*, vol. 27, pp. 45–45, 2010.
[8] J. Bell, Hosting EnterpriseData in the Cloud—Part 9: InvestmentValue Zetta, Tech. Rep., 2010.
[9] Ross, ―Technical perspective: A chilly sense of security,‖ *Commun. ACM*, vol. 52, pp. 90–90, 2009.
[10] D. E. Bell and L. J. LaPadula, Secure Computer Systems: Unified Exposition and Multics Interpretation The MITRE Corporation, Tech. Rep., 1976.
[11] K. J. Biba, Integrity Considerations for Secure Computer Sytems The MITRE Corporation, Tech. Rep.,1977.
[12] H. Harney, A. Colgrove, and P. D. McDaniel,―Principles of policy in secure groups,‖ in *Proc. NDSS*, San Diego, CA, 2001.
[13] P. D. McDaniel and A. Prakash, ―Methods and limitations of security policy reconciliation,‖ in *Proc. IEEE Symp. Security and Privacy*, Berkeley, CA, 2002.
[14] T. Yu and M. Winslett, ―A unified scheme for resource protection in automated trust negotiation,‖ in *Proc. IEEE Symp. Security and Privacy*, Berkeley, CA, 2003.
[15] J. Li, N. Li, and W. H. Winsborough, ―Automated trust negotiation using cryptographic credentials,‖ in *Proc. ACM Conf. Computer and Communications Security (CCS)*, Alexandria, VA, 2005.
[16] V. Goyal, O. Pandey, A. Sahai, and B.Waters,―Attibute-based encryption for fine-grained access control of encrypted data,‖ VA, 2006.
[17] 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*, 2010, pp. 534–542.
[18] J. Bethencourt, A. Sahai, and B. Waters, ―Ciphertextpolicy attributebased encryption,‖ in *Proc. IEEE Symp.*
[19] *Security and Privacy*, Oakland, CA, 2007.R. Bobba, H. Khurana, and M. Prabhakaran, ―Attribute-sets: A practically motivated enhancement to attribute-based encryption,‖ France, 2009.
[20] Sahai and B. Waters, ―Fuzzy identity based encryption,‖ in *Proc. Acvances in Cryptology—Eurocrypt*, 2005, vol. 3494, LNCS, pp. 457–473.