Design and Implementation of Collaborative Ciphertext-Policy Attribute-Role based Encryption for Data Access Control in Cloud

Somchart Fugkeaw, Hiroyuk Sato
University of Tokyo
Tokyo, Japan
somchart@satolab.itc.u-tokyo.ac.jp
schuko@satolab.itc.u-tokyo.ac.jp

ABSTRACT: *In a real-world collaborative data sharing scenario in cloud computing, there are multiple users who can access the resource shared by multiple data owners anytime and anywhere. The evolution of user status, roles, and privilege in the federated data sharing environment become even more and more complex to handle. Efficiently managing multiple access control policies and providing appropriate access control to different groups of user are crucially needed in such collaborative and federated environment. This paper presents the design and implementation of our proposed access control model called Collaborative-Ciphertext Policy-Attribute Role based Encryption (C-CP-ARBE). An administrative tool called CLOUD-CAT is designed and developed to facilitate a flexible, secure, and efficient management of multiple user accesses and multiple access control policies in multi-owner cloud computing environment. CLOUD-CAT is designed and developed based on the integration of Ciphertext Policy-Attribute-based Encryption (CP-ABE) and Role-based Access Control Model (RBAC) access control model. To support policy management, the tool provides secure channel for several data owners to update and administer their access control policies resided at the cloud server. Finally, we present implementation details to demonstrate advanced features and performance analysis of the prototype system.*

Keywords: Collaborative data sharing, Access control, Cloud computing, CP-ABE, RBAC

Received: 15 July 2015, Revised 18 August 2015, Accepted 20 August 2015

1. Introduction

The need of the collaboration among business partners requires the resources (e.g., data) to be shared and accessed by their groups of users. Therefore the data owner generally specifies access control policy to enforce over the resources shared to authorized users with the permissible action. Several web-based groupware are available to support collaboration among users and even to allow users share data connected to their own database by using authentication techniques (such as user and password). In collaborative data sharing in cloud computing, only authentication and general access control policy are not sufficient for an effective data access control.

To illustrate collaborative data sharing in cloud scenario, we use a hospital information system (HIS) as our running example. In HIS, there are several departments that have their own data records and have to share among relying parties. For example, in

patient treatment department, each patient treatment record may be composed of a few related files such as pre-diagnosis file, treatment records. These files are initially encrypted and authorized to the users within the department, related departments, or even external parties such as other partner hospitals, health insurance company, etc. In this case, treatment records need to be accessed by nurse, physician, and doctor with a different privilege (read, write) to deal with the file. In this case the cloud application could provide the access mechanism to and enforce the policies to different users dynamically.

Generally, security and privacy features of existing cloud applications provided by cloud providers focus on authentication options while an authorization is based on access control list or authentication profile of users. The features that support both user access management and policy management (for data owners or administrators) are currently not as well designed as they could be. Especially, the security and privacy preserving for federated environment where multiple data owners outsource their data for the collaboration is essential. The requirements for strong authentication, dynamic authorization, and flexible policy management are paramount importance and they will be addressed by our research solution.

Existing data access control solutions for cloud storage systems generally concentrate on minimizing key management cost, reducing computing cost of interaction between data owner and outsourced data storage, improving scalability, efficient enforcement of a policy etc. There are a few approaches [1,2] that apply traditional cryptographic approaches such as symmetric key encryption and public key encryption for securing data outsourced in the cloud. However, these schemes are not practical if there are a large number of users that use the shared resources. This is because if the symmetric encryption is used, several copied of ciphertext will be generated and key distribution becomes very expensive and risky. For the public key encryption, key management overhead is very high.

Ciphtertext-Policy Attribute-based Encryption (CP-ABE) [3] is recognized as one of the most applicable techniques for cryptographic-based access control for cloud computing since it reduces key management overhead compared to symmetric and public key encryption. In addition, CP-ABE provides the owners with the full control over their own policy. However, in the data sharing scenario, users may obtain their attributes from multiple authorities to gain access to systems. Hence, multi-authority attribute-based encryption (MA-ABE) schemes [4-6, 9, 10, 13] have been proposed.

Nevertheless, CP-ABE possesses the following major limitations. First, CP-ABE generally supports "r*ead access*" by default. CP-ABE access policy specification lacks the details of privilege specification for the enforcement. Second, there is a high subsequent cost for user revocation. If there any user is revoked from the system, the file needs to be re-encrypted and all non-revoked users' keys are required to be updated. This incurs expensive overheads. Finally, access policy is required for data encryption and it is restricted to be performed by data owners only. However, in a real collaborative data sharing, some users may have a need to update the data. Thus, the policy used for data encryption should be selectively shared to users having write privilege. These common problems are still re-current issues requiring further investigation.

The first two problems have been addressed by our previous work [12]. This paper will implement the access control model with a focus on practical deployment. Therefore, the tool architecture design is proposed to satisfy the cryptographic algorithms proposed in [12]. Also, this paper tackles the last problem regarding the privacy preserving policy. Thus, secure policy sharing feature is included in the CLOUD-CAT tool.

Basically, there are following four three requirements pertaining to security and privacy management necessary for the design and development of a collaborative data sharing system in cloud environment.

1.Granting user/password as an authentication mechanism to a large number of users is considered insecure for accessing sensitive data. Besides, since the data is outsourced in the cloud, privacy preserving of data is highly required. Therefore strong authentication and fine-grained with expressive cryptographic-based authorization for serving secure data access control is needed.

2. Outsourcing access control policies in cloud is convincing as it provides flexible policy management and reduces maintenance cost at data owner side. To this end, policy privacy preserving technique must be available and each data owner can securely update the policy anytime and anywhere.

3. Administrative features such as enrolling new users, revoking users and attributes, and managing (creating, updating, and deleting) policies are necessary for supporting multi-data owners and administrators.

In this paper, we design and develop a web-based access control tool to facilitate secure and collaborative data sharing, data access, and policy administration to achieve the above requirements. The tool is implemented based on our proposed access control scheme C-CP-ARBE [12].

The remainder of this paper is structured as follows. Section 2 discusses related research works. Section 3 describes background of CP-ABE, RBAC, and our proposed access control model. Section 4 presents CLOUD-CAT system design and architecture, as well as the policy privacy-preserving technique. Section 5 analyzes security property of our proposed scheme. Section 6 gives detail of the implementation of the prototype. Finally, the conclusion and future work are discussed in Section 7.

2. Related Work

Existing access control solution for data outsourced in cloud computing is usually based on attribute-based encryption (ABE) which is generally classified into two type key-policy attribute-based encryption (KP-ABE) [5,7] and Ciphertext Policy Attribute Based Encryption (CP-ABE) [3]. In KP-ABE, the ciphertext is associated with a set of attributes and user secret key is constructed to associate with the access structure. However, the KP-ABE scheme does not give data owners have a full control over the access policy. In contrast, in CP-ABE, each user is assigned a set of attributes which are embedded into the user's secret key and a public key is defined for each user attribute. The ciphertext is associated with the access policy structure in which the encryptor can define the access policy by her own control. Users are able to decrypt a ciphertext if their attributes satisfy the ciphertext access structure.

To support a more complex environment of cloud computing where there is multi-owner and multi-authority, a multi-authority attribute based encryption (MA-ABE) is recently proposed by several works [4-6, 9, 10, 13, 16]. Nevertheless, these works do not address the policy management issue.

In [4], the authors propose hierarchical attribute-set-based encryption (HASBE) by extending ciphertext-policy attribute-set-based encryption (ASBE) with a hierarchical structure of users. In this scheme, a trusted authority is responsible for generating and distributing system parameters and root master keys as well as authorizing the top-level domain authorities. However, the vulnerability of trusted authority of the hierarchical domains would be at risk to all users. Besides, to serve the revocation and re-encryption process, the data owner needs to be online during the period agreed with users.

Kan Yang et al [6] proposes DAC-MACS (Data Access Control for Multi-Authority Cloud Storage model. The authors apply CP-ABE technique to construct an access control model where there are several multi-authority issuing the attributes. The proposed scheme improves the decryption process and solves revocation problem in ABE by designing the decryption token and key update and ciphertext update algorithms. For the immediate revocation, their scheme reduces cost of data re-encryption since only the ciphertext getting an effect is updated. However, this approach does not support write access and its policy is not applicable for collaborative data sharing.

Concretely, CP-ABE alone does not support collaborative and multi-authority cloud problem because the policy itself suffers from the ability to express the privilege to specific group of users. In addition, key distribution is another major cost for CP-ABE, especially when there are a large number of users. Hence, we address these drawbacks by integrating role-based access control (RBAC) model to construct our proposed access control model, C-CP-ARBE. Our C-CP-ARBE renders zero cost for key distribution as we encrypt all user decryption keys and store them in the cloud. The key management scheme is presented in our previous work [12]. However, the previous work leaves the policy management issue and practical demonstration of the proposed scheme.

Existing cloud applications such as Google Drive, Dropbox, OneDrive, and iCloud can serve functional data outsourcing in general. However, rigorous requirements for strong authentication, fine-grained access control with encryption feature, and flexible authorization for collaborative data sharing are not comprehensively provided by these cloud services. Amazon's EC2 [14] is one of the most popular IaaS services. However, the advanced access models such as RBAC or Attribute-based Access Control (ABAC) are not supported by EC2. It simply restricts the access control by using operating system (OS) image and access control list.

Dongwan Shin et al. [8] proposed a design and implementation framework of a policy-based decentralized authorization tool in cloud computing. There access control architecture is based on XML-based security architecture and RBAC model. However,

the model and tool do not support privacy preserving of the resources shared to users.

In [17], the authors propose the implementation of ciphertext policy-functional encryption (CP-FE) library for supporting data encryption in cloud computing. FE provides relations between secret key and parameters used for data encryption based on CP-ABE. To encrypt the data, the attribute list is required together with the access structure. Nevertheless, maintaining attribute list for every encryption introduces overhead over the naïve CP-ABE. Besides, the library only supports encryption feature while the policy administration is not covered.

According to [18], the security products that can be applied for securing data in cloud computing are discussed. For example, Vaultive offers AES encryption system to encrypt any data leaving the networks. The encryption service can be used to send the secure data to be stored in cloud or any outsourcing server. The company helps people protect cloud-based services like

Office 365 and Exchange. DocTrackr is a security solution that focuses on file-sharing services such as Box and Microsoft SharePoint. DocTrackr provides data sharing features of which the data owner can assign privilege to users they want to share the documents. It also allows the data owner to track the view on the documents. However, these products do not satisfy the full-fledge security and privacy of data outsourced in complex cloud environment where the data needs to be selectively accessed by multiple users.

To the best of our knowledge, this paper is the first attempt to propose the access control tool supporting collaborative data sharing in cloud computing where there are multiple owners sharing resources to multiple users from different organizations. In this paper, we present access control tool called CLOUD-CAT based on an integration of RBAC model and CP-ABE scheme. RBAC is integrated to enable high expressiveness of the cryptographic-based CP-ABE access control. The combined model also offers fine-grained policy enforcement supporting privilege specification and scalable user management. In addition to compliment collaborative access control requirements, we propose a privacy preserving access policy technique to support secure policy sharing for serving data encryption done by authorized users. This feature is also embedded in our CLOUD-CAT.

3. Background

In this section, we describe the concept of CP-ABE, RBAC, and our proposed access control model.

3.1 Ciphertext Policy Attribute-based Encryption (CP-ABE)

Bethencourt et al. [3] propose the ciphertext policy attribute-based encryption as another kind of attribute-based encryption (ABE) for the access control. Basically, the concept of cryptographic construction of CP-ABE is based on the bilinear maps. The following describes the formal definition of bilinear maps.

Definition1
Bilinear maps
Let G_1 and G_2 be two multiplicative cyclic groups of prime order p and e be a bilinear map, $e : G_1 \times G_1 \rightarrow G_2$. Let g be a generator of G_1. Let H: $\{0,1\}^* \rightarrow G_1$ be a hash function that the security model is in random oracle. The bilinear map e has the following properties:

1. Bilinearity: for all $u , v \in G_1$ and $a, b \in Z_p$, $e(u_p, v_p) = e(u, v)^{ab}$.

2. Non-degeneracy: $e(g, g) \neq 1$.

Definition 2: (Access Structure [3]). Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $A \subseteq 2^{\{P1,P2,\ldots,Pn\}}$ is monotone if $\forall$ B,C : if $B \in A$ and $B \subseteq C$ then $C \in A$. An access structure (respectively, monotone access structure) is a collection A of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $A \subseteq 2^{\{P1,P2,\ldots,Pn\}} \setminus \{\emptyset\}$. The sets in A are called the authorized sets, and the sets not in A are called the unauthorized sets.

3.2 Role based Access Control Model (RBAC) [15]

RBAC is an access control model that provides the relationships of user's role , permission, and objects or resources.

Definition 3:

RBAC is a tuple of (*U*, *R*, *P*, *UA*, *PA*, *RH*) where:

(1) *U*, *R*, *P* are given sets that represent the set of users, roles, and permission, respectively.

(2) $UA \subseteq U X R$, a many-to-many user-to-role assignment relations;

(3) $PA \subseteq P X R$, a many-to-many permission-to-role assignment relation;

(4) $RH \subseteq R X R$ is a partial order on R representing the role hierarchy.

3.3 Our Access Control Model

To integrate the RBAC model to enhance the expressiveness and scalability of CP-ABE, we define attribute-role based (A-RBAC) access control model as follows:

Definition4

A-RBAC is a tuple of (*U, R, P, UA, PA, RH*, *attr*) where:

• (U, R, P, UA, PA, RH) is as in the RBAC model

• D: R $\rightarrow 2^{attr}$ is defined as the description of roles by attributes (attr). Here attr←R is identified as a set of attributes D(attr).

In cloud computing environment, we refer User (U) is a person or entity who requests to access with permission P (read or write) the data outsourced by the data owner in the cloud. Attributes(Attr) are a set of attributes used to characterize the user and associated to the particular attribute "*role*". A set of attributes is issued by attribute authority(AA).

Definition 5

Access Control Policy (ACP): Let ACP *T* is a tree that represents the access structure. Each non-leaf node of the ACP tree represents the Role node and threshold gate where the Role node is a parent of threshold gate node. The threshold gate rule is the same as access tree of CP-ABE. We denote the parent of the children node x in the tree by parent(x). Thus, the parent of leaf node x is the pair of {Role node, threshold gate}. The function attr(x) is defined only x is in a leaf node of the tree.

To provide a fine-grained access control, we introduce special attribute "privilege" as an extended leaf (EL) node of the ACP T in order to identify the read or write privilege of the role.

Definition 6

Satisfying an Access Policy

Let an ACP *T* be given and threshold node R as its root node, whose child nodes are role nodes. Each role node has an access tree as its exactly one child. We define its evaluation *E* by the following.

Def. Evaluation:

For each (k_x of num_x) threshold node *k*,

$$\text{Eval}(k) = \begin{cases} 1 \text{ for atleast } k_x \text{ children } c \text{ of numx nodes eval}(c) = 1 \\ 0 \text{ .... otherwise} \end{cases}$$

OR $num_x = 1$ of num_x;

AND $num_x = num_x$ of num_x

For each role *node r*, *eval*(*r*) = *eval* (*r*'s child).

Figure 1 illustrates collaborative data access control for a hospital information system where professional roles: nurse and medical doctor are allowed to access the file resorted at cloud storage. In our access control model, we allow the collaborative users from partner organizations to access shared resources as long as their attributes are specified in the access policy. The policy tree includes privilege information to support practical access control management. Each role contains attributes and privilege for particular data file. Policy is normally managed and administered by data owners.

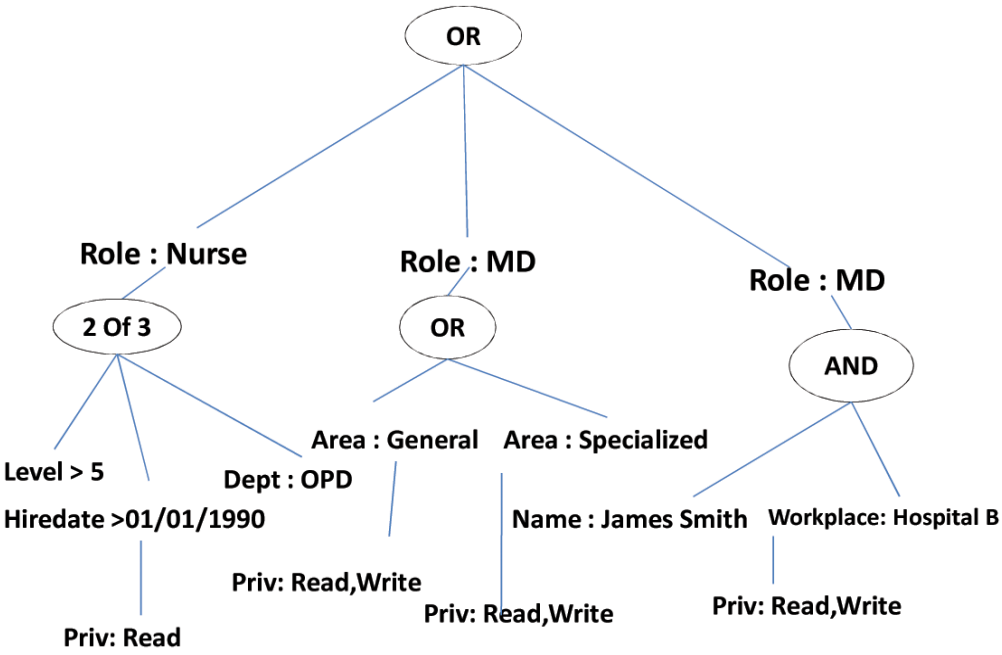

Figure 1. Role Access Policy Structure – OPD(Out- patient department) treatment File

4. CLOUD-CAT System

4.1 System Architecture

This section describes of CLOUD-CAT system architecture and the core techniques of cryptographic algorithms for access control used for the implementation. Figure 2 displays the system architecture.

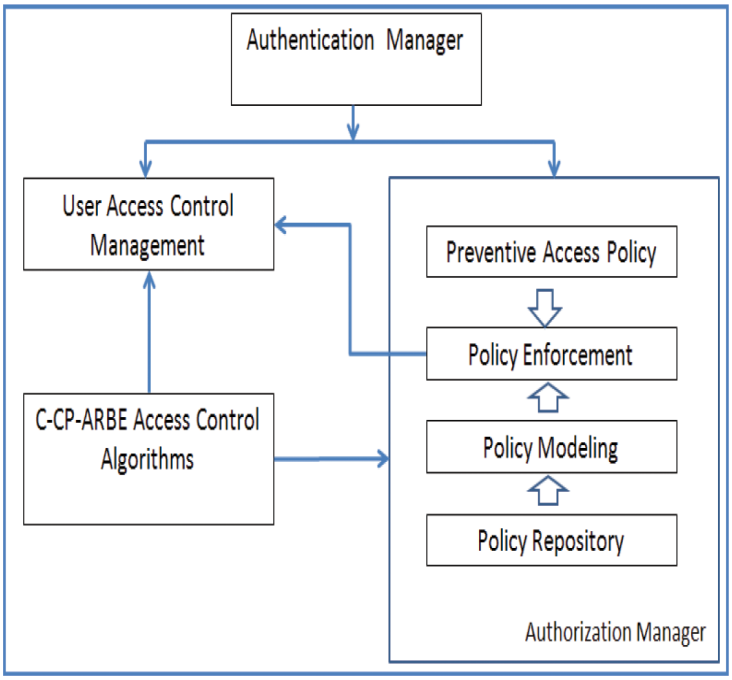

Figure 2. CLOUD-CAT Architecture

The main components of CLOUD-CAT consists of :

1. Authentication Manager
This module is the first gate to control the access of any entities to the data resided in the cloud storage. In our system, users need to create the key pairs and the public key must be signed by the trusted certification authorities (CAs). The users are primarily checked with the details of X.509 certificate including the validity, distinguished name (DN), CA's signature that must be in the certificate trust's list (CTLs), and certificate revocation lists (CRLs).

2. Authorization Manager
This module controls the policy management and enforcement. This module is only served for data owners or administrators who can deploy and manage the policies. It consists of four sub modules.

2.1 Policy Repository
Policy repository is a storage where multiple policies authored by multiple data owners are resided with separate access control.

The stored policy source files are formatted as our specified A-RBAC model. In our system, each policy source file is signed by data owner's private key and it is encrypted by a set of roles and identity attributes of the users who have write privilege. Therefore, only policy authors can manage (add, update, delete) their policies stored in the cloud anytime and anywhere, while authorized users with write privilege can only call the policy for data encryption.

2.2 Policy Modeling
This module is responsible for converting the source policies into ACP tree as it is easy and convenient to visualize policy specifications. The ACP is used by the policy enforcement module. To support other policy models, we exploit the following policy conversion and verification procedure to validate the correctness of policy specification and conform to our defined ACP.

(1) Policy authors specify their policy elements (attributes, roles, privilege) and logical formula of a given policy P.

(2) The procedure assembles the defined policy parts of P into ACP specification.

(3) The procedure runs syntax checking regarding the nodes order and relationship as specified in Definition 4.

(4) The procedure generates verification result (success or fail) to policy authors.

2.3 Policy Enforcement
This module is responsible for enforcing the policy to the users who request for accessing the resources. It supports multi-policy enforcement to multiple users. The policy is enforced based on the access control policies matching the user decryption key. Here, a set of roles with its corresponding attributes determine what privilege the users have over the shared data.

2.4 Preventive Access Policy (PAP)
This module serves an advanced access control feature as similar to IDS-like feature. It consists of a set of pre-defined rule-based policies that specifies illegal access patterns or activities considered as security violations and determines actions to prevent them. Hence, if any illegal access activities are detected, this module will notify the policy enforcement module to perform actions. PAP is stored as a separated file and will be checked and triggered if there are events match the rules.

3. User Access Control Management
This system module provides data access and key retrieval functions to individual user with respect to his/her access right. This module is a center system that is coordinated with other modules for presenting the capability lists to users. If a user is successfully authenticated and her key is matched to any policies encrypting the files, the list of files she has authorized to access will be shown. Then, the encrypted files can be downloaded from the cloud storage.

4. C-CP-ARBE Access Control Algorithms
We exploit a set of access control algorithms we proposed in [10] as an access control mechanism of our tool. This module provides and controls cryptographic processes for user key generation, data encryption and decryption. In this paper, four major algorithms are presented to demonstrate how user and access control is managed behind the tool.

1. **UserKeyGen** ($S_{uid,aid}$, SK_{aid}, $Cert_{uid}$) → $EDK_{uid,aid}$. The KeyGen algorithm takes continuous two steps (1) takes input as set of attributes $S_{uid,aid}$, attribute authority's secret key SK_{aid}, and public key certificate of users $Cert_{uid}$, then it returns the set of user decryption keys UDK (2) a UDK is encrypted with the global public key of the user and outputs the set of encrypted decryption keys $EDK_{uid,aid}$. Then, a set of EDKs will be stored in the user decryption key graph (UDKG) resided in the cloud and it will be assigned to respective users upon the access.

2.**Enc** (PK_{aid}, [SS, GRP] M, *ACP, Cert*$_{uid}$) ⟶ SCT. The encryption algorithm performs two continuous steps as followings:

(1) **Inner Layer:** The algorithm takes as inputs authority public key PK_{aid}, access control policy *ACP*, and data *M*. Then it returns a ciphertext CT.

(2) **Outer Layer:** The algorithm takes group role parameter GRP which is randomly generated from a set of users of all roles. GRP is used as a key together with 3DES algorithm to generate the session key referred as a secret seal SS to encrypt the ciphertext CT. Then, the algorithm returns sealed ciphertext SCT. Finally, the secret seal SS is encrypted with user's public key $Cert_{uid}$, and stored in a cloud server.

3. **Decrypt** (PK_{aid}, *SCT*, GSK_{uid}, EDK_{uid},) ⟶ M. The decryption algorithm performs two continuous steps as follows:

(1) Decrypt the secret seal SS. The algorithm takes user's global secret key GSK_{uid} to decrypt the secret seal encrypted by her public key. Then the algorithm returns the session key and it is used to decrypt the SCT. Then, the CT is obtained.

(2) Decrypt the encrypted decryption key (EDK_{uid}). The algorithm takes user's global secret key GSK_{uid} to decrypt the encrypted key. Then, the algorithm returns the user decryption key UDK. Together the PK_{aid}, if the set of attributes *S* satisfies the ACP structure, the algorithm returns the message M (data file).

4. **RevokeUser**($U_{id,aid}$, SK_{aid} , UL_{Rid}) ⟶ $UL_{Rid'}$, $GRP_{aid'}$. The RevokeUser algorithm takes $User_{uid,aid}$ and AA's secret key, and user list UL of the role having user revoked as inputs. The secret key of attribute authority is used to sign the revoked request and the revoked user is removed from the UL. Then, it returns updated UL. Also, revoked user is deleted from the

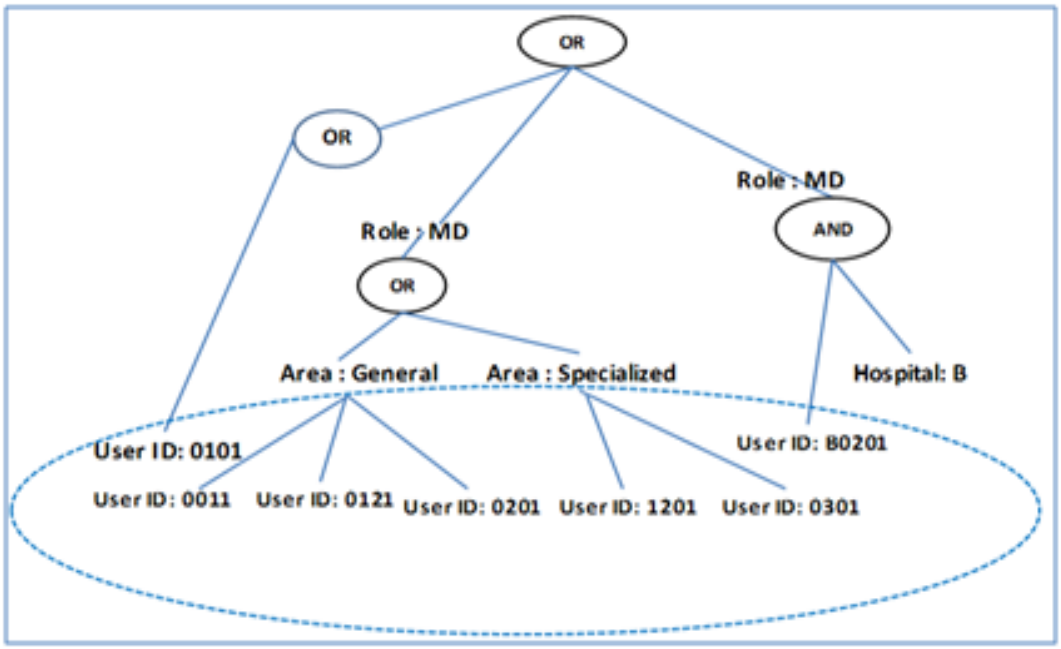

Figure 3. Policy Encryption Policy (PEP)

UDKG where all user encrypted keys are stored. Finally, GRP is updated.

With our revocation scheme, if there is any user revoked from the system, only secret seal is re-generated and it is used to re-encrypt the ciphertext. Thus, all existing user decryption keys are not required to be re-generated.

4.2 Privacy-preserving Access Control Policy

To enable the policies to be stored in cloud to serve the flexible management and immediate use of data encryption performed by users with write permission, we introduce policy encryption policy (PEP) to preserve privacy of the access control policy. To this end, we make use of a simple CP-ABE tree policy to encrypt the ACP. The PEP is simply formed by a set of identity attributes of the data owners and authorized users. Figure 3 shows a sample PEP model.

As can be seen in Figure 4, PEP is constructed by taking all internal nodes (role nodes and all threshold gate nodes) from the normal ACP. For leaf nodes, identity attributes (User ID) of data owners and users who have write access permission are applied. Then, PEP is used to encrypt the ACP. Hence, only data owners and authorized users listed in the PEP are allowed to access the policy and use it to encrypt the data. Since the PEP contains only role attributes and their user identity attributes, the policy encryption and decryption cost are trivial. The proposed PEP technique does not require additional keys for data owners or users with write permission because they can use existing decryption keys (which already contain their identity attributes).

With PEP, a set of access control policies can be stored in a cloud server in an encrypted format, allowing users with write permission to encrypt data anytime. Here, the data owner does not need to provide policy encryption service, which is indeed demanding in real-world data sharing scenarios.

5. Security Analysis

Collusion Resistance: In our scheme, it should be not possible for different users to combine their attribute sets and in order to gain access the resources. Even though the adversaries use an attribute having the identical name and value (position: doctor) from different attribute authorities, they cannot collude with any legal user in the authority sharing the target resource. This is because each attribute has its own public key, unique id together the tag of attribute authority id. Each attribute is also signed by its attribute authority. Based on bilinear map, a random number *r* constituted in each user attribute enables the key is technically distinguishable and non-substitutable. This prevents the collusion attack from the adversaries.

Secure Data Access Control: In addition to strong authentication with X.509 certificate and password, the access control mechanism is supported by a set of C-CP-ARBE algorithms. Users need to have a valid decryption key to access the files. Our proposed model is proven to be secure under general security attack model [10]. As our core model is based on CP-ABE scheme, the security proof of the CP-ABE construct is referred to the details depicted in [3].

Privacy-Preserving Policy: We store the policy in cloud storage to enable flexible policy management. Data owners or administrators can update or deploy policies anytime and anywhere. The privacy of policy is ensured by the public key encryption. All policies resorted at cloud are signed with the data owner's or administrator's public private key. Thus, only data owners who create the policy can update the policy. Users having write privilege can only specify the policy file for data encryption. They also cannot gain access to see the details of any policy files as they are not visualized and presented to the users.

6. Implementation

We implemented CLOUD-CAT using Java and PHP. CLOUD-CAT is a web-based tool to help users and data owners can use the tool flexibly and conveniently. The tool is run on the Apache Sever. For the key management server, we use Open SSL as a core PKI service to generate key pairs to users and system entities. The service is run on E3-1200v3, 4 processors, 3.6 GHz with Ubuntu Linux.

We developed access control policies for a hospital information system. 20 test polices were set up to validate functionality of policy modeling and enforcement of the tool. Over 25 roles from both internal organization and other partner organizations such as partner hospitals and insurance firm are included in the policies. In overall, the tool can dynamically and correctly enforce

constraints and policies to users as well as support administrative functions for data owners/administrators in an efficient manner (comprehensive, correct, and easy-to-use).

6.1 System Functions

6.1.1 Data Access Management for Collaborative Users

Figure 4 shows the screen shot of our web-based access control tool for collaborative users. From this screen, authorized users can access a list of data files shared by multiple owners with respect to their roles and privileges specified in the respective policy. Users can download the encrypted files they authorized to access and use their private key to decrypt the file. Since our system supports multi-owner data sharing, a user may have several decryption keys, the tool facilitate the user to view the user decryption key graph to realize which key is matched to the target file.

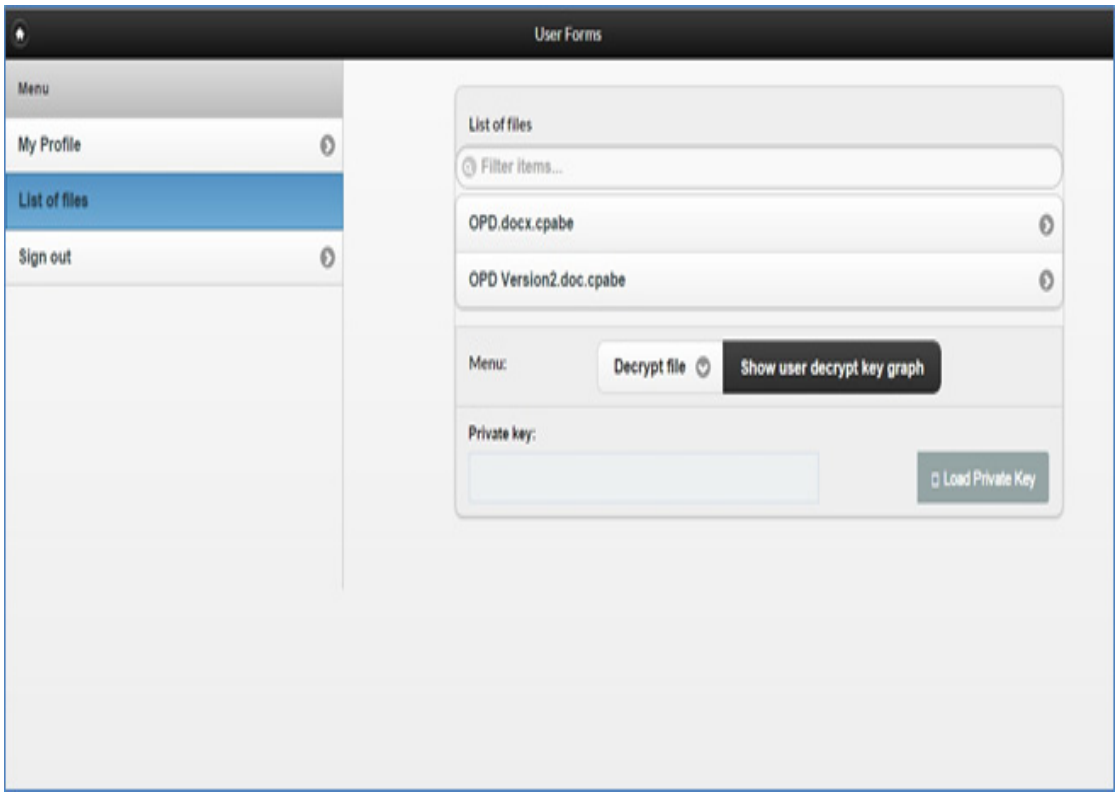

Figure 4. User Access Control Management

6.1.2 Administrative Access Control and Policy Management for Data Owners/Administrators

Figure 5 displays the screen shot for data owners and administrators. Here, administrative features are available to support user-role and role-attribute management (add, update, revoke), data upload, and policy management. For the policy updating, data owners or administrators can edit policy content and add a new preventive access policy. Figure 6 displays the policy management screen where data owners or administrator can flexibly update the policy content via the graphical interface. Administrator can efficiently update policy elements including roles, attributes, privilege, and logical operators without coding effort or naïve policy writing.

6.2 Performance Analysis

6.2.1 Encryption and Decryption Performance

We evaluate the data encryption and decryption performance by measuring the time that the server returns ciphertext for data encryption or plaintext for data decryption. In our experiment, the encryption and decryption time are measured with different file sizes and different number of leaf nodes (attributes) contained in the access control policy. Figure 7 exhibits the encryption and decryption performance regarding the file size. The access control policy used for encryption consists of one attribute.

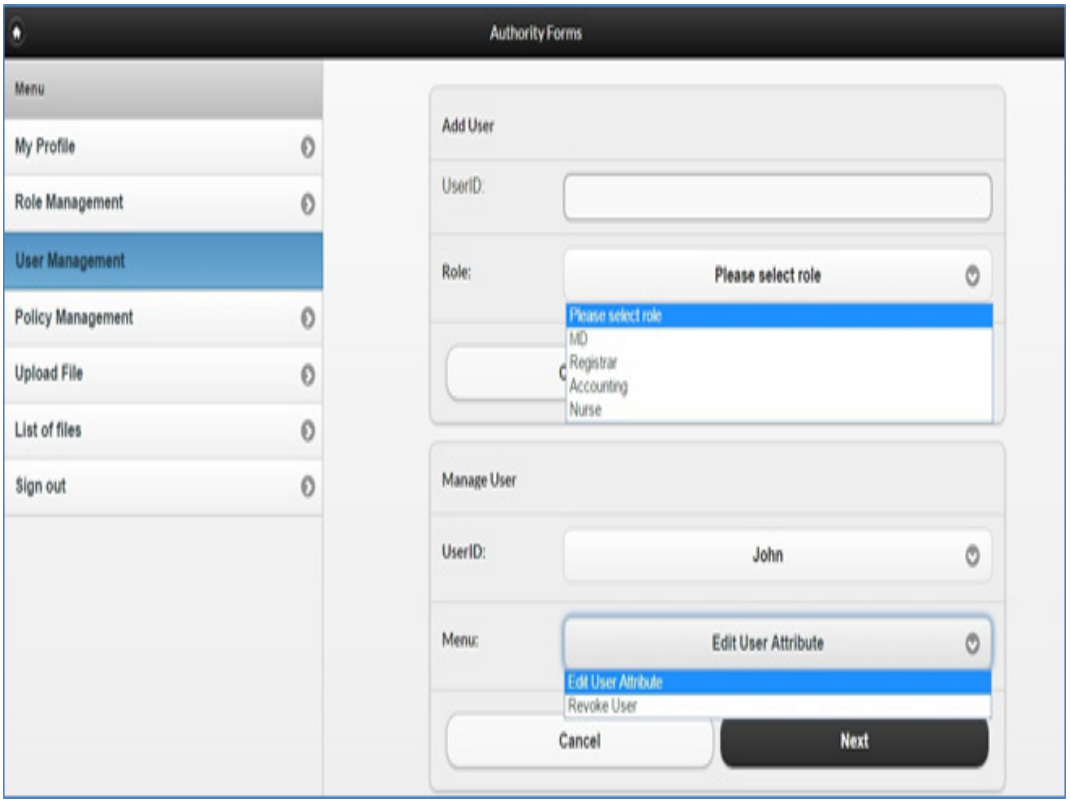

Figure 5. Administrative Function: User Management

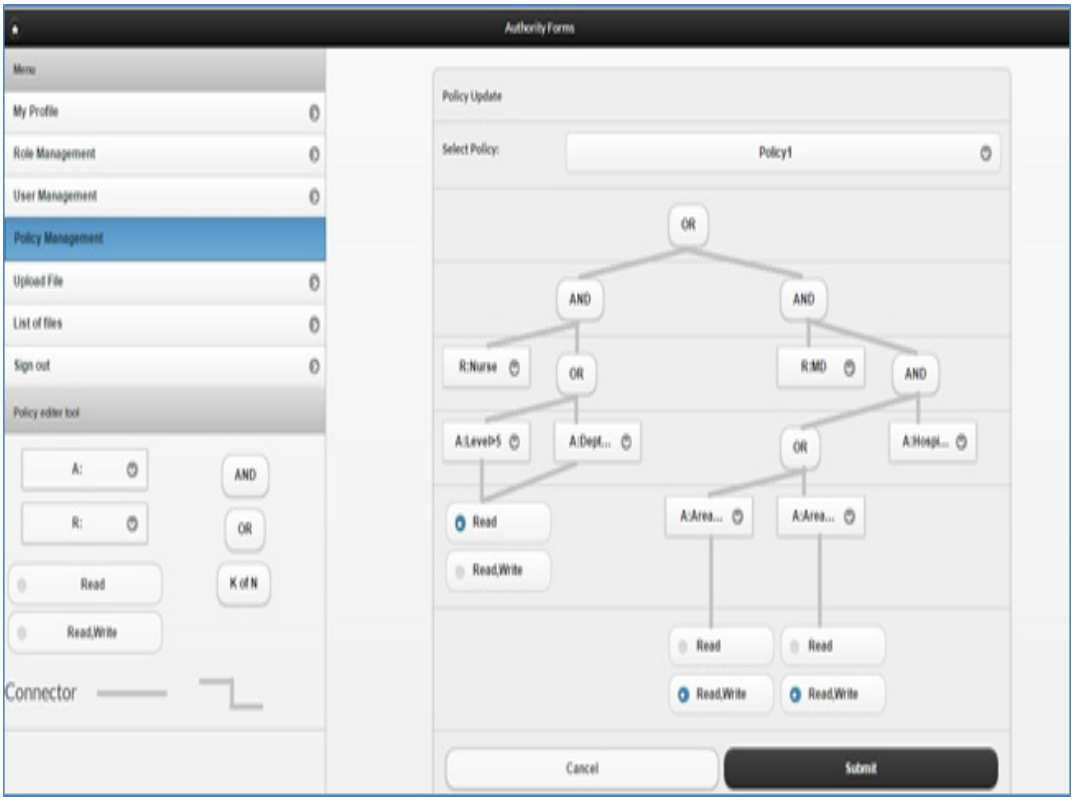

Figure 6. Administrative Function: Policy Update

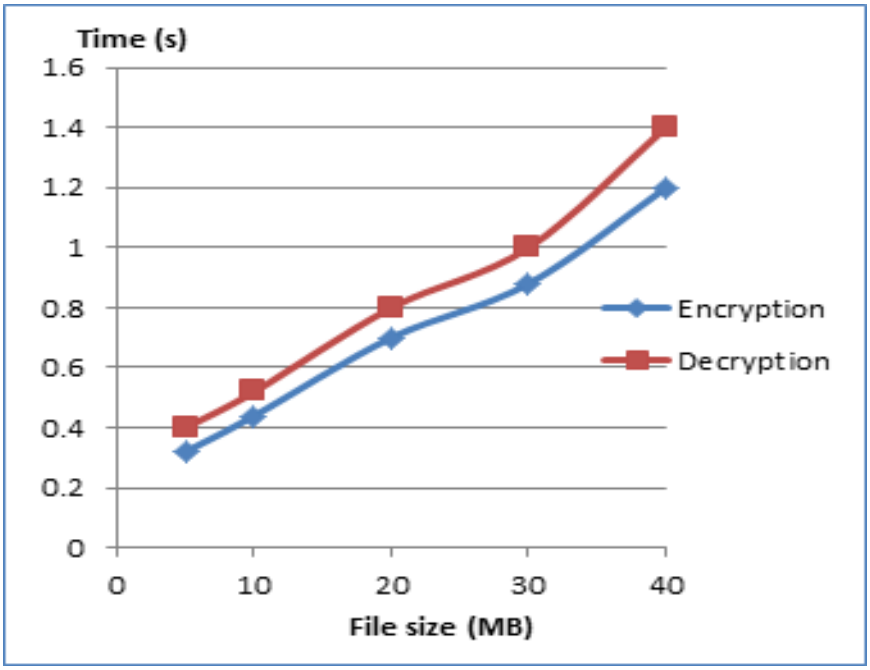

Figure 7. Encryption and Decryption time regarding file size

Figure 8 displays the encryption and decryption time regarding the number of attributes contained in the access control policy, and the file size is 5 MB.

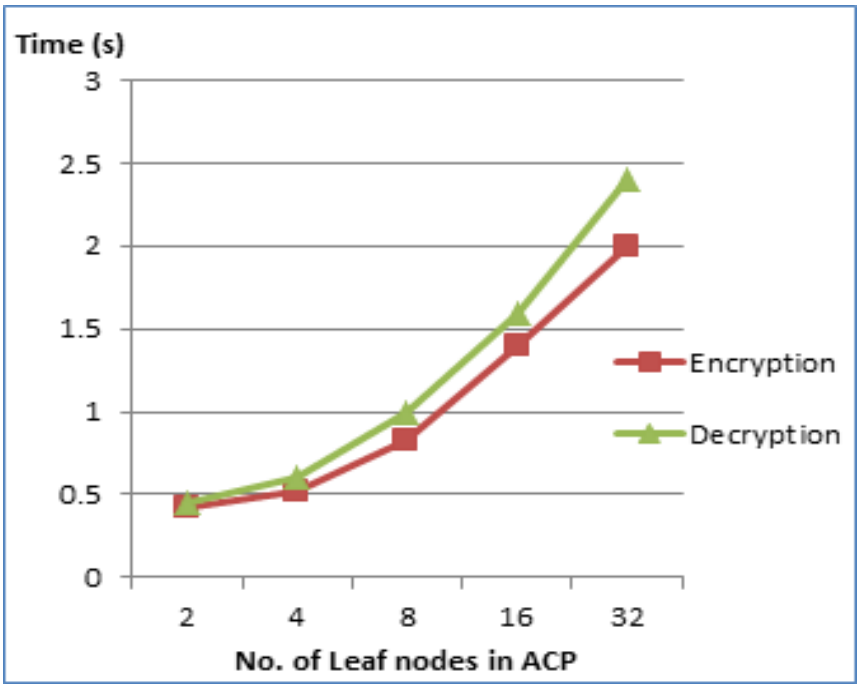

Figure 8. Encryption Time and Decryption Time regarding the number of leaf nodes (attributes) in ACP

The graph shows that encryption and decryption performance is linear to both file size and number of leaf nodes contained in the policy. Even though the size of the plaintext and number of leaf nodes in ACP are quite big, our system reasonably yields the acceptable performance which indicates the practicality for deploying in the real-world cloud systems. We believe that when the proposed system is deployed in the real cloud environment where the superb computing power is in place, the performance is getting much more improved.

6.2.2 Scalability

We evaluate the scalability of the system in terms of server throughput. We measure the time that the server responds to concurrent access requests. The file size of ciphertext is 1 MB and it is encrypted with a policy containing 10 attributes. As of fig. 9, the results shows that response times for supporting requests (decryption by server) were ranging from 12 to 22 requests per second for 100-1,000 concurrent requests. Then, response times tend to continuously decline after 1,000 concurrent accesses.

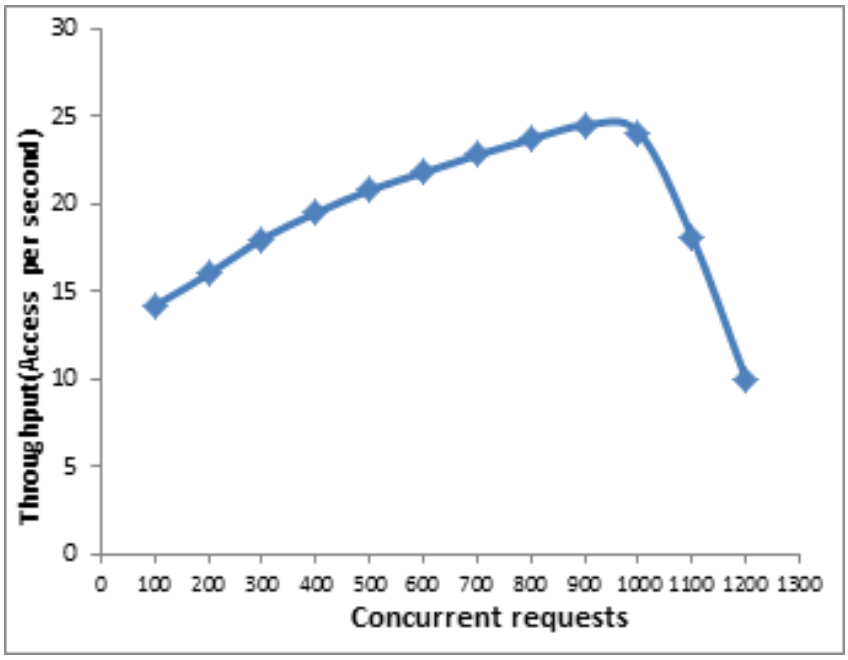

Figure 9. Throughput of Concurrent Accesses

Since we measure the complete access until the plaintext is obtained, server throughput indicates practical and reasonable performance to support a large number of users.

7. Conclusion and Future work

We have presented our web-based access control tool called CLOUD-CAT that provides flexible and secure features for both users and multiple data owners for data sharing in cloud computing. The tool is equipped with necessary features for the collaborative access control supporting secure data sharing and policy management. We also present the privacy preserving access control policy method that enables the efficient management of access control policies stored in the cloud server. For future work, we will extend CLOUD-CAT to support automatic conversion of standard XAMCL access policy. In addition, we will tackle policy conflict detection and resolution and embed this function to enhance policy management function of the tool.

Acknowledgement

This research has been supported in part by Otsuka Scholarship.

References

[1] De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P. (2010). Encryption Policies for Regulating Access to Outsourced Data, *ACM Transactions on Database Systems* (TODS), April.

[2] Mariana Raykova., Hang Zhao., Steven Bellovin. (2012). Privacy Enhanced Access Control for Outsourced Data Sharing, Financial Crypto 2012, Bonaire, March.

[3] Bethencourt, J., Sahai, A., Waters, B. (2007). Ciphertext-policy Attribute-based Encryption, IEEE Symposium of Security and privacy, Oakland, CA, USA, May 20-23, Los Alamitos, 2007.

[4] Zhiguo Wan., Jun-e Liu., Robert, H. Deng. (2012). HASBE: A Hierarchical Attribute-Based Solution for Flexible and Scalable Access Control in Cloud Computing. *IEEE Transactions on Information Forensics and Security* 7 (2) 743-754.

[5] Ming Li., Shucheng Yu., Yao Zheng., Kui Ren., Wenjing Lou. (2012). Scalable and Secure Sharing of Personal Health Records in Cloud Computing using Attribaute-based Encryption, *IEEE Transactions on Parallel and Distributed Systems*.

[6] Yang, Kan., Jia, Xiaohua., Ren, Zhang, KuiBo., Xie, Ruitao (2013). DAC-MACS: Effective Data Access Control for Multiauthority Cloud Storage Systems., *IEEE Transactions on Information Forensics and Security* 8 (11) 1790-1801.

[7] Goyal, V., Pandey, O., Sahai, A., Waters. B. (2006). Attribute-Based Encryption for Fine-grained Access Control of Encrypted Data. *In*: Proceedings of CCS'06, Alexandria, Virginia, USA.

[8] Shin, Dongwan., Wang, Ying., Claycomb,William. (2012). A Policy-based Decentralized Authorization Management Framework for Cloud Computing, *In*: Proceedings of the 27th ACM Symposium on Applied Computing (SAC 12), Riva del Garda (Trento), Italy, March 26-30.

[9] Chase, M., Chow, S. S. M. (2009).Improving privacy and security in multi-authority attribute-based encryption, *In*: Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09), ACM.

[10] Lewko, A. B.,Waters, B. (2011). Decentralizing attribute-based encryption, *In*: Proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology, EUROCRYPT'11, Springer.

[11] Zhou, Lan., Varadharajan, Vijay., Hitchens, Michael. (2013).Achieving Secure Role-Based Access Control on Encrypted Data in Cloud Storage. *IEEE Transactions on Information Forensics and Security* 8 (12) 1947-1960.

[12] Fugkeaw, S., Sato, H. (2015).An extended CP-ABE based Access control model for data outsourced in the cloud, IEEE International Workshop on Middleware for Cyber Security, Cloud Computing and Internetworking (MidCCI 2015), Taichung, Taiwan, July 1-5.

[13] Yang, Kan., Jia, Xiaohua., Ren, Kui., Xie, Ruitao., Huang, Liusheng (2014). Enabling efficient access control with dynamic policy updating for big data in the cloud, *In*: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), Toronto, Canada, April 27 –May 2.

[14] Amazon Elastic Compute Cloud http://aws.amazon.com.

[15] Sandhu, Ravi S., David, Ferraiolo, F. , Kuhn, Richard D. (2000).: The NIST model for role-based access control: towards a unified standard. ACM Workshop on Role-Based Access Control.

[16] Fugkeaw, Somchart. (2015). Achieving privacy and security in multi-owner data outsourcing, *In:* Proceedings of International Conference on Digital Information Management (ICDIM), Macau, August, 2015

[17] Hasegawa, Keisuke., Kanayama, Naoki., Nishide, Takashi., Okamoto, Eiji. (2015). Software Implementation of Ciphertex Policy-Funtional Encryption with Simple Usability, *In*: Proceedings of International Conference on IT Convergence and Security (ICITCS), Malaysia, August.

[18] TrustIT http://www.trustitllc.com/five-cloud-based-security-tools-that-will-protect-your-business-effortlessly/