

Effective Data Access Control for Multi-Authority Cloud Storage with Intrusion Detection

R.Balaji (PG scholar)

Department of Computer Science and Engineering
Sir Issac Newton College of Engg & Tech., Nagapattinam
Anna University: Chennai, TamilNadu, India
rbalaji283@gmail.com

N.Arthi (Assistant Professor)

Department of Computer Science and Engineering
Sir Issac Newton College of Engg & Tech., Nagapattinam
Anna University: Chennai, TamilNadu, India
arthisincet@gmail.com

Abstract—Business Record (BR) is an emerging centric model of information exchange, which is often outsourced to be stored at a third party, such as cloud providers. However, there have been wide privacy concerns as business information could be exposed to those third party servers and to unauthorized parties. To assure the business data control over access to their own BRs, it is a promising method to encrypt the BRs before outsourcing. Yet, issues such as risks of privacy exposure, scalability in key management, flexible access and efficient user revocation, have remained the most important challenges toward achieving fine-grained, cryptographically enforced data access control. To achieve fine-grained and scalable data access control for BRs, we leverage attribute based encryption (ABE) techniques to encrypt all business file. We focus on the multiple data owner scenario, and divide the users in the BR system into multiple security domains that greatly reduces the key management complexity for owners and users. A high degree of data privacy is guaranteed simultaneously by exploiting multi-authority ABE.

Keywords— *Cloud server, business record, multi-authority, cloud storage, attribute revocation*

INTRODUCTION

Automated Business solution resides in many stages of enhancement, which started from standalone application and moved into data centric web application. Globalizing business information makes application more efficient in level of usage. Due to reduce the investment cost and infrastructure maintenance pure web based services are converted into cloud based services. But cloud is dependent in third party investor, who are primary administer to have full control of business information. Due to lack of privacy in business information safety a huge requirement available and should be filled with anonymous data to prevent from any malicious thread. The solution is given for that problem in terms of cryptography.

In last decades, banking process has been globalized and managed by the contracted people in outsourced manner. Bank business information provides many number of feasibilities in accessing customers records like account transaction, and history in one place through the web, which has made the storage, retrieval, and sharing of the business information more efficient. Mostly, customer information is accessed in from data centric view by using various user interfaces. Due to the high cost of building and maintaining specialized data centers, many banking services are outsourced to or provided by third-party service providers, for example, Cognizant, like these corporation. Recently, architectures of storing banking

information in cloud computing have been proposed in many terms.

Even it exists in more convenient for customer, many notable security and privacy risk factors available. In cloud based service, globalizing business information is easier for everyone, but more doubtful on reliability on information security. Because third party service provider for infrastructure or environment is lack in securing the privacy about data. Even though number of security last year's resides of every tier of application, the third-party storage servers are often the targets of various malicious behaviors which may lead to exposure of the potential customer information. As a famous incident, a firm in Milwaukee, includes confidential material gathered from 420,000 websites, including household names, and small Internet sites. To ensure customer-centric privacy control over their own Banking Business Information, it is essential to have fine-grained data access control mechanisms that work with semi-trusted servers.

A feasible and promising approach would be to encrypt the data before outsourcing. Basically, the Bank owner herself should decide how to encrypt her files and to allow which set of users to obtain access to each file. A Banking file should only be available to the users who are given the corresponding decryption key, while remain confidential to the rest of users. Furthermore, the bank customer shall always retain the right to not only grant, but also revoke access privileges when they feel it is necessary. However, the goal of data-centric privacy is often in conflict with scalability in a Banking system. The authorized users may either need to access their own information for personal use or professional purposes. Examples of the former are family member and friends, while the latter can be bank professional, advisor, and researchers, etc. We refer to the two categories of users as *personal* and *professional* users, respectively. The latter has potentially large scale; should each owner herself be directly responsible for managing all the professional users, she will easily be overwhelmed by the key management overhead. In addition, since those users' access requests are generally unpredictable, it is difficult for an owner to determine a list of them. On the other hand, different from the single data owner scenario considered in most of the existing works in a Banking system, there are *multiple owners* who may encrypt according to their own ways, possibly using different sets of cryptographic keys. Letting each user obtain keys from every owner whose records she wants to read would limit the accessibility since customers are not always online. An alternative is to employ a central authority (CA) to do the key management on behalf of all record owners, but this requires too much trust on a single authority (i.e., cause the key escrow problem). In this paper,

we endeavor to study the data centric, secure sharing of banking information stored on semi-trusted servers, and focus on addressing the complicated key management issues. In order to protect the personal data stored on a semi-trusted server.

We adopt attribute-based encryption (ABE) as the main encryption primitive. Using ABE, access policies are expressed based on the attributes of users or data, which enables a customer to selectively share her records among a set of users by encrypting the file under a set of attributes, without the need to know a complete list of users. The complexities per encryption, key generation and decryption are only linear with the number of attributes involved. However, to integrate ABE into a large-scale banking system, important issues such as key management scalability, dynamic policy updates, and efficient on-demand revocation are non-trivial to solve, and remain largely open up-to-date. To this end, we make the following main contributions:

(1) We propose a novel ABE-based framework for data-centric secure sharing of banking process in cloud computing environments, under the multi-owner settings. To address the key management challenges, we conceptually divide the users in the system into two types of domains, namely *public* and *personal domains*. In particular, the majority professional users are managed distributively by attribute authorities in the former, while each owner only needs to manage the keys of a small number of users in her personal domain. In this way, our framework can simultaneously handle different types of information sharing applications' requirements, while incurring minimal key management overhead for both owners and users in the system. In addition, the framework enforces write access control, handles dynamic policy updates, and provides break-glass access to Banking information's under emergence scenarios.

(2) In the public domain, we use multi-authority ABE (MA-ABE) to improve the security and avoid key escrow problem. Each attribute authority (AA) in it governs a disjoint subset of user role attributes, while none of them alone is able to control the security of the whole system. We propose mechanisms for key distribution and encryption so that banking business record owners can specify personalized fine-grained role-based access policies during file encryption. In the personal domain, owners directly assign access privileges

for personal users and encrypt a file under its data attributes. Furthermore, we enhance MA-ABE by putting forward an efficient and on-demand user/attribute revocation scheme, and prove its security under standard security assumptions. In this way, customers have full privacy control over their records.

(3) We provide a thorough analysis of the complexity and scalability of our proposed secured information sharing solution, in terms of multiple metrics in computation, communication, storage and key management. We also compare our scheme to several previous ones in complexity, scalability and security. Furthermore, we demonstrate the efficiency of our scheme by implementing it on a modern workstation and performing experiments/simulations.

Compared with the preliminary version of this paper there are several main additional contributions:

- We clarify and extend our usage of MA-ABE in the public domain, and formally show how and which types of user-defined file access policies are realized.
- We clarify the proposed revocable MA-ABE scheme, and provide a formal security proof for it.

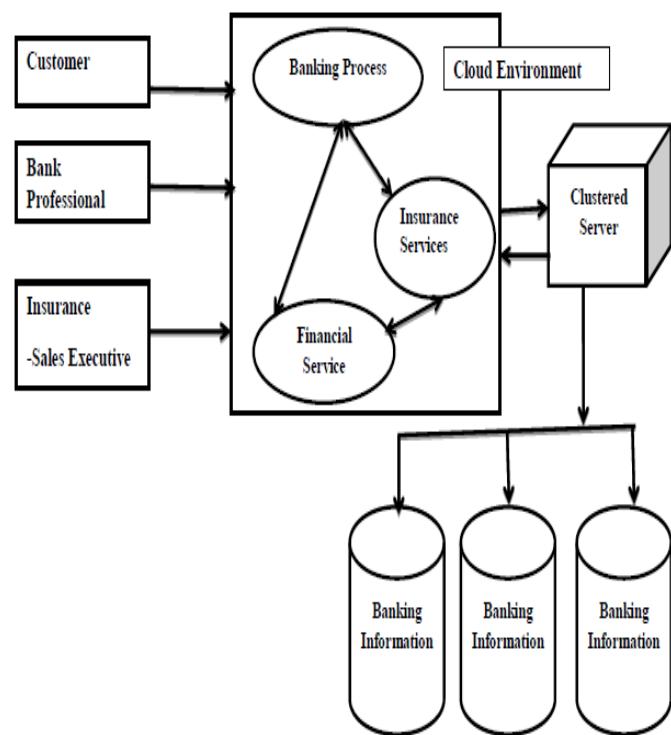
We carry out both real-world experiments and simulations to evaluate the performance of the proposed solution in this paper. In multi-authority cloud storage systems, user's attributes can be changed dynamically. A user may be entitled some new attributes or revoked some current attributes. And his permission of data access should be changed accordingly. However, existing attribute revocation methods [9] either rely on a trusted server or lack of efficiency, they are not suitable for dealing with the attribute revocation problem in data access control of multi-authority cloud storage systems.

Attribute based encryption is implemented based on AES technique, which is an encryption algorithm for securing sensitive but unclassified material by U.S. Government agencies and, as a likely consequence, may eventually become the de facto encryption standard for commercial transactions in the private sector. It is used to provide encrypt and decrypt the attribute values of user records.

SYSTEM MODEL AND SECURITY MODEL

A. System Model

We consider a data access control system in multi-authority cloud storage, as described in Fig. 1. There are five types of entities in the system: a certificate authority (CA), attribute authorities (AAs), data owners (owners), the cloud server (server) and data consumers (users). The CA is a global trusted certificate authority in the system. It sets up the system and accepts the registration of all the users and AAs in the system. For each legal user in the system, the CA assigns a global unique user identity to it and also generates a global public key for this user. However, the CA is not involved in any attribute management and the creation of secret keys that are associated with attributes. For example, the CA can be the Social Security Administration, an independent agency of the United States government. Each user will be issued a Social Security Number (SSN) as its global identity. Every AA is an independent attribute authority that is responsible for entitling and revoking user's attributes according to their role or identity in its domain. In our scheme, every attribute is associated with a single AA, but each AA can manage an arbitrary number of attributes. Every AA has full control over the structure and semantics of its attributes. Each AA is responsible for generating a public attribute key for each



attribute it manages and a secret key for each user reflecting their attributes.

Figure 1 System model of data access control in Multi-authority Cloud Environment

Each user has a global identity in the system. A user may be entitled a set of attributes which may come from multiple attribute authorities. The user will receive a secret key associated with its attributes entitled by the corresponding attribute authorities. Each owner first divides the data into several components according to the logic granularities and encrypts each data component with different content keys by using symmetric encryption techniques. Then, the owner defines the access policies over attributes from multiple attribute authorities and encrypts the content keys under the policies. Then, the owner sends the encrypted data to the cloud server together with the ciphertexts.² they do not rely on the server to do data access control. But, the access control

happens inside the cryptography. That is only when the user's attributes satisfy the access policy defined in the cipher text; the user is able to decrypt the cipher text. Thus, users with different attributes can decrypt different number of content keys and thus obtain different granularities of information from the same data.

B. Frame work

The framework of our data access control scheme is defined as follows.

Definition 1 (Framework of Multi-Authority Access Control Scheme). The framework of data access control scheme for multi-authority cloud storage systems contains the following phases:

1) System Initialization- we consider the server to be semi-trusted, i.e., honest but curious as those in [28] and [15]. That means the server will try to find out as much secret information in the stored BR files as possible, but they will honestly follow the protocol in general. On the other hand, some users will also try to access the files beyond their privileges. For example, a pharmacy may want to obtain the prescriptions of patients for marketing and boosting its profits. To do so, they may collude with other users, or even with the server. In addition, we assume each party in our system is preloaded with a public/private key pair, and entity authentication can be done by traditional challenge-response protocols.and not as an independent document. Please do not revise any of the current designations.

2) Key Generator- The Key Generator used to generate the key for encryption based on available preferred techniques. AES will produce compact keys with the additional benefit that the cryptosystem is not burdened with patent compliance. However, should a binary fall to reverse Engineering, the key will become compromised (note that AES is a Symmetric Cipher - not an Asymmetric Cipher which has Public and Private Keys). Currently, there are three FIPS (Federal Information Processing Standards) approved symmetric encryption algorithms: AES, Triple DES, and Skipjack. This article will use AES or the Advanced Encryption Standard in CBC Mode. Note that DES (FIPS 46-3) was withdrawn in May 2005, and is no longer approved for Federal use. AES (or Rijndael - pronounced "Rhine dahl") is the work of Joan Daemen and Vincent Rijmen - hence the portmanteau Rijndael. AES is a 128 bit block cipher that accepts key lengths of 128, 192, and 256 bits. The required number of rounds (i.e., linear and non-linear transformations), depend on the key size. Below are the FIPS 197 conformant Key-Block-Round-Combinations.

Taking from FIPS 197:

For both its Cipher and Inverse Cipher, the AES algorithm uses a round function that is composed of four different byte-oriented transformations: 1) Byte substitution using a substitution table (S-box), 2) Shifting rows of the State array by different offsets, 3) Mixing the data within each column of the State array, and 4) Adding a Round Key to the State.

3) Data encryption by Owners- The main goal of our framework is to provide secure user-centric BR access and efficient key management at the same time. The key idea is to divide the system into multiple security domains (namely,

public domains (PUDs) and personal domains (PSDs) according to the different users' data access requirements. The

	Key Length (Nk words)	Block Size (Nb words)	Number of Rounds (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Key-Block-Round Combinations

PUDs consist of users who make access based on their professional roles, such as doctors, nurses and medical researchers. In practice, a PUD can be mapped to an independent sector in the society, such as the health care, government or insurance sector. For each PSD, its users are personally associated with a data owner (such as family members or close friends), and they make accesses to BRs based on access rights assigned by the owner.

Each data owner is a trusted authority of her own PSD, who uses a KP-ABE system to manage the secret keys and access rights of users in her PSD. Since the users are personally known by the BR owner, to realize user centric access, the owner is at the best position to grant user access privileges on a case-by-case basis. For PSD, data attributes are defined which refer to the intrinsic properties of the BR data, such as the category of a BR file. For the purpose of PSD access, each BR file is labeled with its data attributes, while the key size is only linear with the number of file categories a user can access. Since the number of users in a PSD is often small, it reduces the burden for the owner. When encrypting the data for PSD, all that the owner needs to know is the intrinsic data properties.

4) Data encryption by Users- In our framework, there are multiple SDs, multiple owners, multiple AAs, and multiple users. In addition, two ABE systems are involved. We term the users having read and write access as data readers and contributors, respectively. The owners upload ABE-encrypted BR files to the server. Each owner's BR file is encrypted both under a certain fine grained and role-based access policy for users from the PUD to access, and under a selected set of data attributes that allows access from users in the PSD. Only authorized users can decrypt the BR files, excluding the server.

5) Intrusion alert- In proposed system, intrusion alert system takes major responsibility to alert anonymous access thread to relevant user / Authority control. The module has well designed rich user interface to obtain the number attempt and it will assume thread when the attempt exceed maximum trial attempt. The attempt will be counted as one of try in user login/ data access process even in case if any wrong entry by authenticated user. Authentication process of data owner and customer will be logged by cloud server logger. For any customer/ user account, account status will be disabled by setting status flag of user access control list. Login attempt and host details will be logged in non-trustable host list. Decision about access host is made by getting and analyze about its certificate status. Commonly trusted parties will have always authorized signature in certificate.

6) Attribute Revocation- User status is revoked to earlier state, when it is disabled by human error or in used by others. User revocation is processed in two stages. In first step,

disabled/ noted user will get message about its authentication fail on allowed trines. User has to verify their identity to prove the uniqueness. Once the identity is confirmed new access code or secret key will be send by data owner from the server. user has to prove some account information about their account to identify the current user is trustable. New secret key will be generated cloud server and forward to user.

C. Security Model

In multi-authority cloud storage systems, we make the following assumptions:

The CA is fully trusted in the system. It will not collude with any user, but it should be prevented from decrypting any cipher texts by itself. . Each AA is trusted but can be corrupted by the adversary. The server is curious but honest. It is curious about the content of the encrypted data or the received message, but will execute correctly the task assigned by each attribute authority. Each user is dishonest and may collude to obtain unauthorized access to data.

All of the cryptographic algorithms we have looked at so far have some problem. The earlier ciphers can be broken with ease on modern computation systems. The DES algorithm was broken in 1998 using a system that cost about \$250,000. It was also far too slow in software as it was developed for mid-1970's hardware and does not produce efficient software code. Triple DES on the other hand, has three times as many rounds as DES and is correspondingly slower. As well as this, the 64 bit block size of triple DES and DES is not very efficient and is questionable when it comes to security. What was required was a brand new encryption algorithm that would be resistant to all known attacks. The National Institute of Standards and Technology (NIST) wanted to help in the creation of a new standard. However, because of the controversy that went with the DES algorithm, and the years of some branches of the U.S. government trying everything they could to hinder deployment of secure cryptography this was likely to raise strong scepticism. The problem was that NIST did actually want to help create a new excellent encryption standard but they could not get involved directly. Unfortunately they were really the only ones with the technical reputation and resources to lead the effort.

Instead of designing or helping to design a cipher, what they did instead was to set up a contest in which anyone in the world could take part. The contest was announced on the 2nd of January 1997 and the idea was to develop a new encryption algorithm that would be used for protecting sensitive, non-classified, U.S. government information. The ciphers had to meet a lot of requirements and the whole design had to be fully documented (unlike the DES cipher). Once the candidate algorithms had been submitted, several years of scrutinisation in the form of cryptographic conferences took place. In the first round of the competition 15 algorithms were accepted and this was narrowed to 5 in the second round. The fifteen algorithms are shown in table 7 of which the 5 that were selected are shown in bold. The algorithms were tested for efficiency and security both by some of the world's best publicly renowned cryptographers and NIST itself.

1) Inner Workings of a Round- The algorithm begins with

$$\begin{aligned}
 \mathbf{s}^0 & 0,j = (2 \cdot s_{0,j}) \oplus (3 \cdot s_{1,j}) \oplus s_{2,j} \oplus s_{3,j} \\
 \mathbf{s}^0 & 1,j = s_{0,j} \oplus (2 \cdot s_{1,j}) \oplus (3 \cdot s_{2,j}) \oplus s_{3,j} \\
 \mathbf{s}^0 & 2,j = s_{0,j} \oplus s_{1,j} \oplus (2 \cdot s_{2,j}) \oplus (3 \cdot s_{3,j}) \\
 \mathbf{s}^0 & 3,j = (3 \cdot s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \cdot s_{3,j})
 \end{aligned}$$

an Add round key stage followed by 9 rounds of four stages and a tenth round of three stages. This applies for both encryption and decryption with the exception that each stage of a round the decryption algorithm is the inverse of its counterpart in the encryption algorithm. The four stages are as follows:

1. Substitute bytes
2. Shift rows
3. Mix Columns
4. Add Round Key

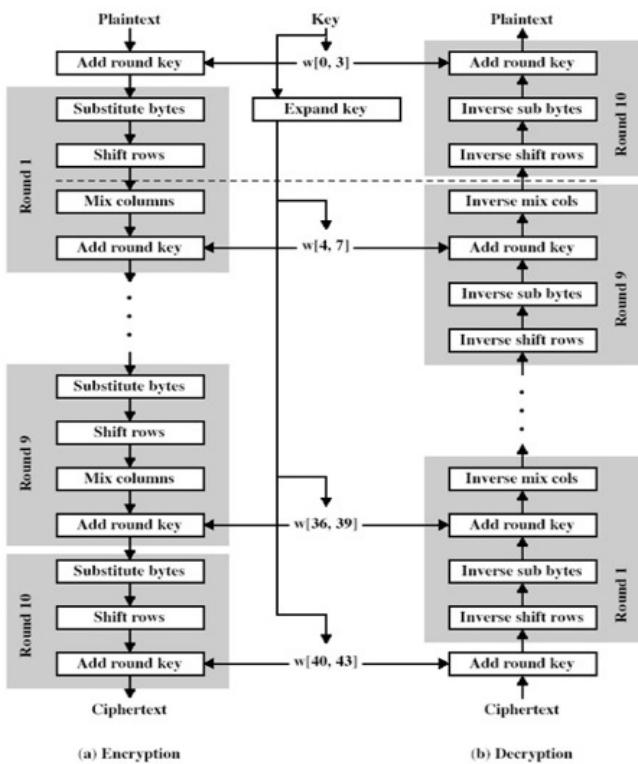


Figure 2 Workings of a round key stage

The tenth round simply leaves out the Mix Columns stage. The first nine rounds of the decryption algorithm consist of the following:

1. Inverse Shift rows
2. Inverse Substitute bytes
3. Inverse Add Round Key
4. Inverse Mix Columns

Again, the tenth round simply leaves out the Inverse Mix Columns stage. Each of these stages will now be considered in more detail.

2) MixColumn Transformation

This stage (known as MixColumn) is basically a substitution but it makes use of arithmetic of GF(28). Each column is operated on individually. Each byte of a column is mapped into a new value that is a function of all four bytes in the column. The transformation can be determined by the following matrix multiplication on state:

Equation 1

Where \bullet denotes multiplication over the finite field GF(28).

Figure 3 MixColumns stage

As an example, let's take the first column of a matrix to be $s_{0,0} = \{87\}$, $s_{1,0} = \{6E\}$, $s_{2,0} = \{46\}$, $s_{3,0} = \{A6\}$. This would mean that $s_{0,0} = \{87\}$ gets mapped to the value $s_0 = \{47\}$ which can be seen by working out the first line of equation 1 with $j = 0$. Therefore we have:

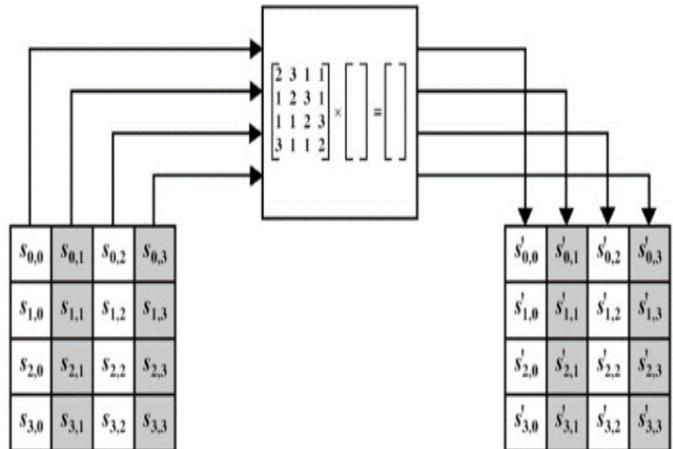
$$(02 \bullet 87) \oplus (03 \bullet 6E) \oplus 46 \oplus A6 = 47$$

So to show this is the case we can represent each Hex number by a polynomial:

$$\{02\} = x$$

$$\{87\} = x^7 + x^2 + x + 1$$

Multiply these two together and we get:



$$x \bullet (x^7 + x^2 + x + 1) = x^8 + x^3 + x^2 + x$$

The degree of this result is greater than 7 so we have to reduce it modulo an irreducible polynomial $m(x)$. The designers of AES chose $m(x) = x^8 + x^4 + x^3 + x + 1$. So it can be seen that

$$(x^8 + x^3 + x^2 + x) \bmod (x^8 + x^4 + x^3 + x + 1) = x^4 + x^2 + 1$$

This is equal to 0001 0101 in binary. This method can be used to work out the other terms. The result is therefore:

00010101

10110010

01000110

$$\begin{array}{r} \oplus 10100110 \\ \hline 01000111 \end{array} = \{47\}$$

This first matrix of equation 1 can be shown to be the inverse of the first matrix in equation 7.3. If we label these A and A-1 respectively and we label state before the mix columns operation as S and after as S0, we can see that: AS = S0 therefore, A-1S0 = A-1AS = S

CONCLUSION

The revocable multi-authority system has been implemented based on CP-ABE as the security criteria in data access control of the cloud storage. The storage system architectural environment resides like multi-stage or multi-authority based data access on business information. To reduce any security block whole, the system ensure the authentication by using attribute encryption rather than traditional data access as anonymously. The difficulties of revocation are reduced while applying attribute-based encryption by enabling multi-level independent authority for every level of data owner. Where data owner is the primary controller of the data and allows for requesting user to provide any attribute key for ensure reliable access. Middle level component reside for every access in Cloud data access. Rich level user interface is available to gather all information on authentication. Key attributes of a level has been verified and pool of data access from the cloud server is provided. Central data storage being

as cloud data center and provides data access on request based on verification defined by data owner. Controller model has the component to take effective process on incoming requests. The multi-authority CP-ABE is a promising technique, which can be applied in any remote storage systems.

Acknowledgment

We would like to thank Head Ms.S.Nithyadhevi, and Ms.N.Arathi, Asst.Professor for their supports, suggestions and the reviewers for helpful comments.

References

- [1] Bethencourt. J, Sahai .A, and Waters. B, “Ciphertext-Policy Attribute-Based Encryption,” in Proc, 2007.
- [2] Bhavani Thuraisingham, Mohamed Nabeel, Elisa Bartino, “Towards Privacy Preserving Access Control in the cloud” in Proc, 2011.
- [3] Hemanthi .K, Sree Bala .M, Sai Sathyanarayana .S, “Multi-authority ABE for Securely Sharing PHRs in Cloud Computing” in Proc, 2014.
- [4] Pooja K.Patil, Pawar .P.M, “PHR Model using Cloud Computing and Attribute-Based Encryption”, in Proc, 2013.
- [5] Tejaswini .M and Roopa .C.K, Ayesha Taranum, “Securing Cloud Server and Data Access with Multi-Authorities,” in Proc, 2014
- [6] Thomas Hupperich, Hans Lohr, “Flexible Patient-Controlled Security for Electronic Health Records,” in Proc, 2010.
- [7] Waters. B, “Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization,” in Proc, 2011.
- [8] Xiaohui Liang, and Rongxing Lu, “Ciphertext Policy Attribute Based Encryption with Efficient Revocation,” in Proc, 2011.
- [9] Xiang-Yang Li, Taeho Jung, Zhiguo Wan and Meng Wan, “Privacy Preserving Cloud Data Access with Multi-Authorities”, in Proc, 2013.
- [10] Ximeng Liu, Jianfeng Ma, Jinbo Xiong, and Guangjun Liu, “Ciphertext-Policy Hierarchical Attribute-based Encryption for Fine-Grained Access Control of Encryption Data”, in Proc, 2014.
- [11] Yang .K and Jia .X, “Attribute-Based Access Control for Multi-Authority Systems in Cloud Storage,” in Proc, 2012.
- [12] Yi Mu, Jinguang Han, Willy Susilo, “Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption”, in Proc, 2012.
- [13] Yu .S. Wang .C, Ren, K and W. Lou, “Attribute Based Data Sharing with Attribute Revocation,” in Proc, 2010.
- [14] Zheng .Y, Ren .K, and Lou .W, “Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption,” in Proc, 2013.