2015 IEEE 39th Annual International Computers, Software & Applications Conference

An Extended CP-ABE based Access Control Model for Data Outsourced in the Cloud

Somchart Fugkeaw[1] Hiroyuki SATO[2]
Department of Electrical Engineering and Information Systems
University of Tokyo
somchart@satolab.itc.u-tokyo.ac.jp[1], schuko@satolab.itc.u-tokyo.ac.jp[2]

Abstract—This paper proposes an access control scheme called Collaborative Ciphertext-Policy Attribute Role Based Encryption (C-CP-ARBE). Our C-CP-ARBE integrates Role-based Access Control (RBAC) into a Ciphertext-Policy Attribute-based Encryption (CP-ABE). The proposed model provides high expressiveness of access control policy, scalable user management, and less user revocation cost compared to the existing approach. In addition, our model supports both read and write access control in a more complex data sharing in collaborative cloud storage where there are multi-owner, multi-user, and multi-authority. For the evaluation, we develop the access control tool and set up test cases to validate the functionality of our proposed scheme. We also conduct the performance evaluation and compare the revocation cost of our C-CP-ARBE and CP-ABE scheme to demonstrate that our revocation method incurs less computation cost and efficient in practice for supporting a larger scale of users.

Keywords- acceess control; privacy; collaborative cloud; key management; user revocation; attribute-based encryption

I. 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. In collaborative data sharing in cloud computing, only authentication and general access control policy are not sufficient for an effective data access control. This is because cloud storage server is considered as an "honest but curious" [1] or semi-trusted servers where the data owner cannot fully trust them. In addition, the users and cloud service providers are not in the same security domain and sensitive data is in risk for its privacy and security for being hacked, modified, and disclosed.

For this reason, outsourced data is generally encrypted and the collaborative users must have a key(s) to decrypt with respect to the access control policy enforcement. Nevertheless, preserving confidentiality and privacy become more subtle for outsourced data shared among multiple groups of users across different domains. First, an access control policy must be flexible, expressive and able to enforce different data access permissions over the multiple groups of user from collaborative parties. Second, the access control must be scalable in supporting a large number of users from different organizations. Third, user revocation cost leading to re-key generation of non-revoked users and file re-encryption must be minimized.

To describe the collaborative data sharing scenario, we use a hospital information system (HIS) as our running example. In HIS, all data from all departments are encrypted and stored at a cloud storage. It is assumed that each department is the owner of the data generated by their staffs. In a treatment department, each patient treatment record may compose 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, the access control policy must be enforced with different privileges to all authorized users regardless of the organizations they belong to. Also, users such as a doctor may have a key that can decrypt files generated in both her hospital and other partner hospitals she is granted the right. In case of user revocation, if there is any user resigns or changes the position, revoking such user should not incur the effects to exciting users.

Nevertheless, no other exiting approaches have been shown to address the access control problem in the collaborative and cross domain data sharing in a cloud. As a result, devising a collaborative access control model for multi-owner and multi-authority cloud storage that is capable to support *flexible, fine-grained, and scalable access control*, and *efficient user revocation scheme* is a real challenge.

In this paper, we propose a novel collaborative access control scheme called C-CP-ARBE based on the integration of role-based access control (RBAC) model and CP-ABE scheme.

Our proposed model possesses the following contributions:

1. Our proposed C-CP-ARBE integrate RBAC model to render flexible and fine-grained access control enforcement. The proposed scheme also provides the efficient management of a large number of users based on roles-attributes assignment. This enables our scheme is highly scalable.

2. We propose an efficient user revocation scheme providing free cost for data re-encryption cost and the re-key generation of existing non-revoked users.

3. We provide the implementation that consists of the development of access control tool and performance evaluation that compares the user revocation cost of our proposed scheme and traditional CP-ABE.

0730-3157/15 $31.00 © 2015 IEEE
DOI 10.1109/COMPSAC.2015.216

The remainder of the paper is structured as follows. Section 2 reviews related works. Section 3 gives definitions used in our proposed access control model, describes C-CP-ARBE model, and analyzes key challenges of access control. Section 4 gives details about the experiment. Finally, section 5 concludes the paper.

II. Related work

Attribute-based encryption (ABE) is regarded as a suitable solution for formulating a light-weight access control to outsourced data. The key construction of attribute-based encryption is based on bilinear maps. Goyal et al. proposed key-policy attribute-based encryption (KP-ABE) [6] to serve a more general and richer encrypted access control. In this scheme, the ciphertext is associated with a set of attributes for each of which a public key component is defined. User secret key is constructed to associate with the access structure. However, the KP-ABE-based schemes [2,4] do not give the data owner has a full control over the access policy.

To address this drawback, the Ciphertext Policy Attribute Based Encryption (CP-ABE) was proposed in [10]. In CP-ABE, 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.

In [11], the authors proposed a role-based encryption (RBE) scheme for cloud storage systems. The proposed RBE uses ABE for cryptographic access control and use identity broadcast encryption for key distribution. For the access control, the role-based access control (RBAC) policy is enforced through a public parameter of role and a group public to encrypt the data. However, in this system data is encrypted by the data owner to the specific role, several copies of the encrypted data are required for users in different roles.

To support multi-owner and multi-authority cloud, a multi-authority attribute based encryption (MA-ABE) is recently proposed by several works [3-5,8,9, 12].

In [3], 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. Each user in the system is assigned a key structure which specifies the attributes associated with the user's decryption key. However, the vulnerability of trusted authority of the hierarchical domains would be at risk to all users.

Kan Yang et al [5] 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 the cost for 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.

III. C-CP-ARBE Model

In this section, we provide the definitions of the access control elements, system model, security proof, and the analysis of our C-CP-ARBE.

3.1 System Definitions

Definition1: **User (U), Role (R), Attributes (attr), and Permission (P)**

We are given a set of users (U).We denote by (R) the set of roles of which each user is assigned to a particular role. In our setting, we assume that a set of attributes (attr) are assigned to each user by the attribute authority (AA).

Furthermore, we consider permissions as read(r)/write(w) given by a resource (data files) owner.

Definition 2: Access Control Policy (ACP)

ACP is a monotone tree-based structure used to represent a set of attributes (attr) that belongs to a specific role (r). ACP expresses the roles and attributes relationship and logical conditions by using AND, OR, or k of n threshold gate. ACP is constructed from a set of roles and attributes administered by the respective attribute authority (AA).

ACP tree (represented as *T*) is expressed by the following elements.

- Let $R(r_1, r_2, \ldots r_n)$ be a set of roles
- Let S_k $(attr_1, attr_2, \ldots, attr_n)$ be a set of attributes
- Let φ_T denote the set of all the non-leaf nodes in the tree *T*
- Let Y be the set of real leaf-nodes in *T*
- Let Priv(Priv:Read, Priv:Write) be the privilege for the access that has the value either (1) read or (2) write.

Example 1: Let a set of roles {nurse, MD} be given. 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.

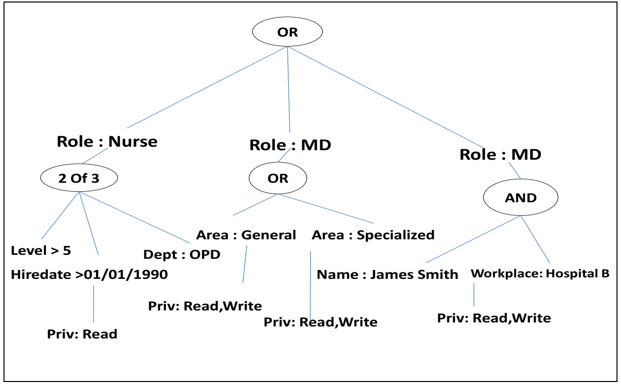


Figure 1: Role Access Policy Structure – Out Patient Data (OPD) treatment (data sharing between two domains)

According to the above policy, there are two major roles nurse, and medical doctor from two hospitals are allowed to access the OPD file with different privileges. Under each specific role, there are a set of attributes characterizing the role to satisfy the policy conditions.

3.2 System Model

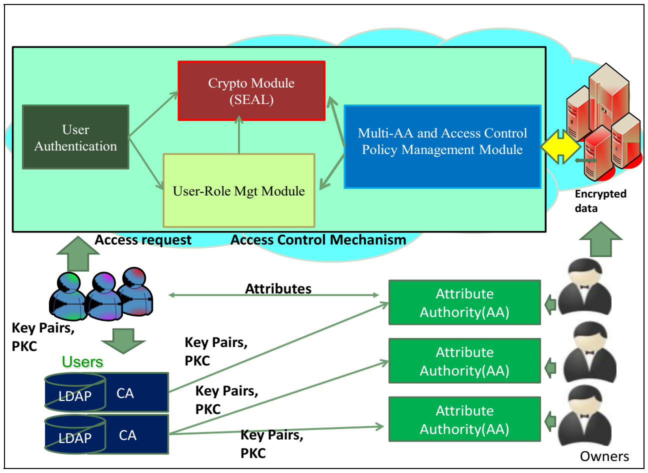


Figure 2: Collaborative Access Control in Multi-Authority Cloud Storage Systems

Figure 2 illustrates our system model which consists of the following four entities and our access control mechanism.

1. Attribute Authorities (AA) is the independent parties who issue, revoke, and update users' attributes according their roles of the particular domain. Each AA is responsible for generating public attribute keys for all attributes belong to the AA and issuing the secret keys to users enrolled in the domain.
2. Certificate Authorities (CA) is the trusted parties who issue the public key certificate (X.509 certificate) to all entities including users, AAs, data owners, system agents.
3. Users are the entities that request to access (read or write) the data outsourced in the cloud. Each user is assigned the set of attributes with respect to his/her role by the attribute authority.
4. Data owners initially upload their data in the encrypted form to the cloud server. They also specify the access control policy to regulate how the users gain access the particular resource and what privilege they have over the resource.

Our access control mechanism composes of the following four main system modules which are resorted at the cloud.

1. Authentication Module (AM)

This module is the first gate to control the access of any entities to the data resided in the cloud storage. The users are primarily check with the details of X.509 certificate including the validity, distinguished name, CA's signature that must be in the certificate trust's list (CTLs), and certificate revocation lists (CRLs).

2. User-Role Management Module (URMM)

In this module, all authorized users are registered and mapped to the role. Under the role, attributes associated to users are grouped to describe the role of the users. This module maintains the role and attributes constructs. Public role parameter of each role is generated to represent as a public value shared among the users of the role. The public role parameter is changed when there is any revocation of the member of the role and a new version of public role parameter is created.

3. Crypto Module (CM)

The Crpyto module is a core engine of our proposed scheme. At a nutshell of this model, we propose two-encryption layer called SEAL (Secret Encryption over Attribute-based Encryption Layer) to support strong encryption and enable the optimization for key management and user revocation cost. This module provides and controls a cryptographic process for user key generation, data encryption and decryption, and revocation. These algorithmic details will be given in next section.

4. Multi-AA and Access Control Policy Management Module (MACPM)

This module controls multiple attributes issued by multiple attribute authorities (AAs). Each AA also has its own public key, private key and certificates obtained from trusted CAs. The access control policy of each AA is encrypted by the public key of the data owner and only data owner can update and access the content of the policy.

3.3 C-CP-ARBE Cryptographic Model

Our proposed cryptographic process of C-CP-ARBE scheme is based on the bilinear map used in traditional ABE [6]. We describe the construction of our model as follows.

Phase 1: System Setup

This phase consists of four algorithms as follows:

1. **Create Attribute Authority**(AA_{id},)→PK_{aid}, SK_{aid}, $PK_{x.aid}$. The algorithm takes the attribute authority ID(AA_{id}) as input. It outputs the authority public key (public parameter) PK_{aid}, SecretKey SK_{aid}, and public attribute keys $PK_{x.aid}$ for all attributes issued by the AA_{aid}. AA has also a key pairs ($PubK_{aid}$, $PrivK_{aid}$) generated by CA.
2. **UserRegister**(U_{id}, $Cert_{uid.caSignature}$)→UL'. The userRegister algorithm takes input as userID and user's certificate issued by a trusted CA. If the user is authorized, the user list associated to particular role is updated.
3. **CreatRole**(SK_{aid},R_{ID}, *Set of* U_{id})→MK_R,UL. The CreateRole algorithm takes as inputs attribute

authority's secret key SK_{aid} , RoleID R_{ID}, and set of users U_{id} who belong to the role. It returns master key of role MK_R, and user list UL.

4. **Create GroupRole parameter**($PrivK_{aaid}$,set of R_{ID})$\rightarrow$GRP. The Create GroupRole parameter algorithm takes input as a set of R_{ID} and returns the GRP. Then, the GRP is signed (encrypted) by AA's private key and it will be updated when there is any adding or revoking of user to/from any role.
5. **CreateUDKG**(set of $U_{id,aid}$)$\rightarrow$UDKG. The algorithm takes set of user who uses the resource in the authority domain. It outputs the user decryption key graph (UDKG) with the root node labelled as the $user_{id}$ and empty key node. UDKG is a graph structure used to store the UDKs encrypted by each user's private key. Hence, all decryption keys are not distributed to users but they are stored in a cloud.

Phase 2: Key Generation

This phase consists of two algorithms as follows:

6. **UserKeyGen**($Su_{id,aid}$, SK_{aid}, $Cert_{uid}$)$\rightarrow EDK_{uid,aid}$. The KeyGen algorithm takes continuous two steps (1) takes input as set of attributes $Su_{id,aid}$, 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 $EDK_{uid,aid.}$
7. **UpdateUDKG**(U_{id}, $EDK_{uid,aid}$)$\rightarrow$UDKG'. This algorithm takes user id U_{id} and encrypted decryption key (EDK) to update the UDKG.

Phase 3: Encryption

This phase runs our two encryption layer protocol which accommodates the following two algorithms:

8. **Enc**(PK_{aid} {SS, GRP} M, ACP) $\rightarrow$CT. 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 GRP and generates 3DES session key as a secret seal SS to encrypt the ciphertext CT. It returns sealed ciphertext SCT.

9. **EncSeal**(SS, $Cert_{uid,aaid}$)$\rightarrow$ $ESS_{vid,uid,aid}$ The algorithm takes inputs as secret seal SS and then encrypts the SS with the $Cert_{uid}$ and publishes the encrypted secret seal (ESS) to the user decryption key graph UDKG and stored in pair with the $EDK_{vid,uid,aid}$.

Phase 4: Decryption

10. **Decrypt** (PK_{aid}, SCT, GSK_{uid}, EDK_{uid},) $\rightarrow$ 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} and then obtains the session key to decrypt the SCT and get the CT.

   (2) Decrypt the encrypted decryption key (EDK_{uid}). The algorithm takes user's global secret key GSK_{uid} and then obtains the user decryption key UDK. Together the PK_{aid}, if the set of attribute S satisfies the ACP structure, the algorithm returns the message M.

Phase 5: Revocation

To revoke the users, there are two following algorithms:

11. **RevokeUser**($U_{id,aid}$, SK_{aid} , UL_{Rid})$\rightarrow UL_{Rid'}$, $GRP_{aid'}$. 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. Finally, GRP is updated.
12. **ReEncSS**($AdminPrivk_{aid}$, GRP_{aid}', new SessionKey, SS)$\rightarrow$ SS',SCT'. The algorithm first requires data owner or administrator's private key $AdminPrivk_{aid}$ to decrypt the existing SS. Then, the updated GRP and new session key are constructed to re-encrypt the CT. The algorithm returns an updated secret seal SS' and updated sealed ciphertext SCT'.

3.4 Security Model

Definition 3: Our C-CP-ARBE scheme is secure under security model based on the Decisional Bilinear Diffie-Hellman Problem (DBDH) in the following security game.

Setup. The simulator receives a key pair from a CA. For uncorrupted authorities in S'_A - S_A, the challenger runs CreateAttributeAuthority algorithm and gives a public keys *PK* to the adversary *A*. For corrupted authorities $S'_{A,}$ the challenger sends both the public keys and secret keys to adversary.

Phase1: The adversary submits (($Su_{id,aid}$), $Cert_{uid}$) to the challenger, where ($Su_{id,aid}$) is a set of attributes belonging to an uncorrupted authority AA_{id}. The challenger gives the corresponding user decryption keys UDK to the adversary.

Challenge. Adversary A submits two challenge messages m0 and m1 to the simulator. The simulator flips a fair binary coin ν, and returns an encryption of mν. The ciphertext is computed as follows:

CT = (ACP, $\hat{C} = m_{\nu} Z$, $CT = h^{s}$,
$\forall y \in Y\colon C_y = g^{q_y}(0)$, $C'_y = H(att(y))^{q_y(0)}$). Where γ is a chosen set of attributes. If μ= 0 then $Z = e(g,g)^{\alpha s \cdot}$.

Therefore, the ciphertext is a valid random encryption of message m_{ν}. Otherwise, if μ= 1 then $Z = e(g,g)^{z \cdot}$. We then have $\hat{C}$ = mν $e(g,g)^{z \cdot}$. Since z is random, $\hat{C}$ will be a random element of G_2 from the adversaries view and the message contains no information about m_{ν}.

Phase 2. The simulator performs as it did in Phase 1.
Guess. Adversary *A* submits a guess of *ν' of ν.* The advantage of *A* in this game is defined as Pr[$\nu' = \nu \mid \mu = 0$]=$\frac{1}{2}$

3.5 An Analysis of our proposed scheme

We analyze our proposed scheme against the security requirements as follows.

- *Flexible, fine-grained, and scalable access control*

  We integrate RBAC model into CP-ABE to provide a more expressiveness of policy specification. In the policy tree structure, the operations AND, OR, and K of N are supported to logically express the natural evaluation for roles and attributes as the access control rules. The policy also accommodates the privilege (read or write) of users for each role distinctively. User attributes from multiple domains can be specified under the respective policy of any data owners. We demonstrate the efficiency and practicality of the access control features through the functionality evaluation.

  Regarding the scalability, the proposed model enables CP-ABE policy tree to support a more large number of users and better attribute management by assigning a group of attributes belong to the specific role. This enables the model is scalable in terms of multiple user management.

  In addition, we exploit user decryption key graph (UDKG) to make all user decryption keys are securely stored in a cloud. User keys will be dynamically invoked upon the user's request for access. This provides zero cost for key distribution and enables efficient multiple keys assignment and retrieval. This is a desired feature that could make our proposed model is practical in a large scale of data sharing environment where there is multi-user, multi-owner, and multi-authority. In contrast, approaches based on CP-ABE require distributing every user decryption keys to all individual users who request for the key. The cost for key delivery therefore depends on network conditions and is linear to the number of registered users.

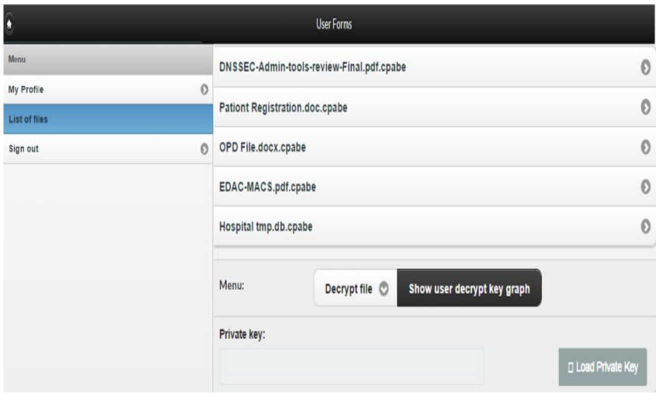
- *Efficient user revocation*

  Based on our encryption technique, the ciphertext produced from the data encryption layer is encrypted by the secret seal (SS) computed from shared role parameter. Since the SS is a symmetric 3-DES key, its generation process is very fast. For user revocation, if there is any user revocation request, only the SS needs to be updated and it will be used to re-encrypt the ciphertext and produce a new sealed ciphertext. To this end, a revoked user cannot use their existing secret seals (SS) to decrypt the cipertext as their keys and certificates are no longer valid the PKI system. We present how our revocation scheme is efficient and consumes minimal cost compared to the CP-ABE in the performance evaluation of revocation cost.

As the analysis of the above distinct features, our scheme is able to address major limitations of existing works using CP-ABE scheme [3-5, 8, 9, 12] by enabling a more expressiveness(flexibility and fine-grainedness) and high scalability of access control as well as optimized user revocation cost. As of our knowledge, our scheme is a novel solution designed to support access control in a collaborative and cross-domain data sharing which has not been addressed by existing works.

IV. EXPERIMENTAL EVALUATION

We simulate a cloud storage server and we develop a web-based access control application and our core crypto module by using PHP and Java language and it 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 Intel(R)Xeon(R)-CPU E5520, 2.27GHz with Ubuntu Linux.



DNSSEC-Admin-tools-review-Final.pdf.cpabe
Pationt Registration.doc.cpabe
OPD File.docx.cpabe
EDAC-MACS.pdf.cpabe
Hospital tmp.db.cpabe

Figure 3: Access Control System: User Form

Figure 3 displays a user form that shows the list of encrypted files the accessing user has the right to decrypt. The user must download the encrypted user decryption key and use his/her private key to locally decrypt the UDK to further decrypt the file.

4.1 System Functionality Evaluation

To assess the system functionality of our C-CP-ARBE, we design collaborative access control policies (ACPs) allowing multi-user in different domain authorities access to the outsourced data files. 20 test cases with various ACPs and sample users from different domains (having set of attributes from several AAs) are set up to validate the correctness of the proposed algorithms and policy enforcements.

For the evaluation, we check that shared files encrypted by the policies that contain multiple roles and set of attributes issued by other AAs can be decrypted by any users (from any domains) who hold the keys satisfying the respective ACP. Also, the privilege of user access is also checked whether it corresponds to the one that is specified in the ACP.

The test result reveals that all test cases perfectly pass the validation as all users can decrypt and encrypt (having write access privilege) the file as specified in all access control policies specified by collaborative owners.

This confirms that the proposed access control is functionally correct as the policy is correctly enforced to the all access cases as well as the proposed key management algorithms(key generation, zero key broadcast, data encryption, and decryption) can perform their functions correctly with the acceptable performance.

We also simulate user revocation cases and validate the revocation algorithm that the revoked user cannot use the key to access the system. The results show that no revoked users are able to use their existing keys to access the files they ever decrypt. This confirms the correctness and reliability of our proposed algorithms.

4.2 Evaluation of User Revocation Cost

We evaluate the performance of the user revocation cost to demonstrate the effectiveness of the revocation scheme of our C-CP-ARBE. In the test scenario, we use a set of policies that contains different number of roles and attributes. The user revocation cost is evaluated by the total time required to complete the revocation process.

Figure 4 depicts an example of the revocation test case where the policy that contains 5 roles and 20 attribute-leaf nodes is used to encrypt a 1 MB file. We compare the revocation cost of our C-CP-ARBE and CP-ABE encrypted with the diffident no. of users accessing files.

According to the revocation performance, our C-CP-ARBE provides significant improvement for the user revocation process over the traditional CP-ABE. This is because our scheme does not require re-key generation, key distribution, and file re-encryption while these operation costs for CP-ABE are linear to the number of remaining users.

Therefore, our scheme is highly practical to be deployed in a real-world revocable data access control environment.

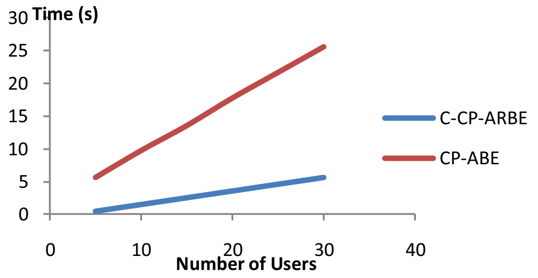


Figure 4: User Revocation Cost

V. Conclusion

We have presented our proposed access control scheme C-CP-ARBE which is based on the combination of CP-ABE and Role-based access control model. The proposed scheme achieves the scalable and efficient collaborative data sharing in multi-authority cloud data storage systems. Specifically, our two-layer encryption strategy provides a significant improvement for key distribution and user revocation.

References

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

[2] Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou, "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing," **IEEE INFOCOM** 2010, San Diego, CA, March, 2010.

[3] Zhiguo Wan, Jun-e Liu, Robert H. Deng: 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, 2012.

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

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

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

[7] M. Chase, **"Multi-authority attribute based encryption"**, in Proceedings of the 4th Theory of Cryptography Conference on Theory of Cryptography (TCC'07), Springer, 2007.

[8] M. Chase and S. S. M. Chow, "**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, 2009.

[9] A. B. Lewko and B. Waters, **"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 2011.

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

[11] L. Zhou, V. Varadharajan, and M. Hitchens, **Enforcing Role-based Access Control for Secure Data Storage in the Cloud**, The Computer Journal, Vol. 54 No.10, 2011.

[12] S.Fugkeaw, **Achieving privacy and security in multi-owner data outsourcing**, IEEE International Conference on Digital and Information Management (ICDIM 2012), Macau, August 2012.