A Secure Authorization System in PHR based on CP-ABE

Ho Hui Chung[1], Peter Shaojui Wang[1], Te-Wei Ho[2], Hsu-Chun Hsiao[1], Feipei Lai[123]

[1]Department of Computer Science and Information Engineering,
[2]Graduate Institute of Biomedical Electronics and Bioinformatics,
[3]Department of Electrical Engineering,
National Taiwan University, Taipei, Taiwan

Abstract—*In this paper, we proposed a method based on the Ciphertext-Policy Attribute-Based Encryption (CP-ABE) to dynamically authorize the user according to their private keys. The key feature of our model is the users do not have to involve in key revocation process. Our model utilizes an additional Key Management Server (KMS) which uses the lazy revocation technique and additional attribute keys to perform effective user revocation. It enables the cloud to differentiate user authentication sessions to protect their keys from different session attackers and this approach could perform direct user revocations from a group without user interventions. The operation does not require re-encryption of existing ciphertext. Our method supports backward and perfect forward secrecy and is escrow-free. Lastly, we present that our method is secure under the chosen identity key attack.*

Keywords: Session Authorization, Attribute Keys, Direct Revocation

I. INTRODUCTION

At present, the hospital starts to share patients' information and electronic medical records online for higher availability of their health records. It is also better for patients on treatment in different hospitals. Such system (namely the Personal Health Record, PHR) benefits from medical records availability for patients receiving treatment in different hospitals. The PHR is commonly presented these days due to the advancement in today's cloud system, which serves as the successor of EHR (Electronic Health Records) which stores data in local hospital. PHR serves as a medium to provide doctors with patient records across different medical institutions using the cloud system which resulting in new security challenges in securing medical records made online. A number of CP-ABE PHR schemes are proposed in contrast to KP-ABE [1] due to it is more "owner centric" in terms of document security. However, these medical records are still under the threat from increasing unknown parties' interceptions.

Therefore, in order to secure documents while lowering the overhead of servers, the concept of Attribute-Based Encryption (ABE) [2] is discovered and slowly put into practice. This method incorporates the access permissions as part of the ciphertext to prevent malicious intentions of permission modification thereafter. There are two well-known versions of ABE such as the Key-Policy Attribute-Based Encryption (KP-ABE) [1] and the Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [3]. The CP-ABE version of ciphertext has a built-in access policy where users have to hold a set of descriptive attribute keys to fulfill the document policy in order to recover the document decryption key.

From the latest data security perspective, the cloud storage systems should be providing data owner with the following security specifications: (1) Dynamic access rights management; (2) Fine-grained access control; (3) Efficient key management; (4) Efficient data encryption; (5) Document secrecy; and (6) User-friendly.

II. METHODS

A. Concept

This paper proposed a new approach, namely the Dynamic Key Update and Delegation (DKUD) in CP-ABE. The DKUD algorithm will perform an update on the user's attribute keys from the key management server (KMS) storage upon successfully login. The KMS runs the key update algorithm that uses the user login session key to perform the attribute keys update or revocation, and the resulting updated key is only used to the particular logged in user because he owns the sessions. The user with a private key registered under his identity key (IDK) could then make use of the identity key IDK to request the key encryption key (KEK) of a document from the cloud. The cloud generates the KEK by user's request using the document delegation key (DK) from the cloud storage and identity key of the user. The user with a combination of a private key, attribute keys and KEK could recover the document encryption key (DEK) to decrypt the encrypted document.

Since the users do not own the actual attribute keys, the system could perform direct revocations when necessary as opposed to [4, 5, and 6]. Key renewal process for unrelated users are effectively made obsolete in this technique where revocation of user attributes would not trigger key updates for the other users sharing the same attributes as in [7, 8]. The system could easily modify the user session key expiration to

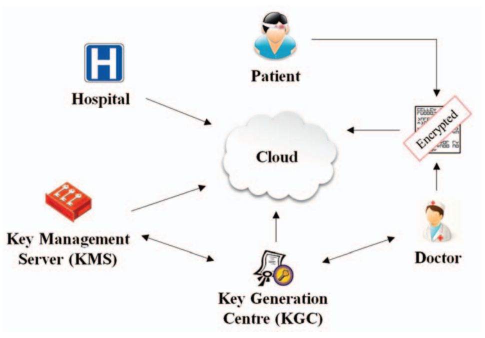


Fig. 1. DKUD-based Secure Personal Health Record System

reduce the key update overhead with a little tradeoff in security. It is still significantly efficient compared to [9].

B. *Dynamic Key Update and Delegation (DKUD)*

The first part of the scheme explains the scheme protocol and assumptions as well as the architecture entities, which will make use of to achieve the goal described in the above section. The second part of the scheme gives the details of algorithms, which are being used by the DKUD. The scheme of Figure 1 introduces an additional entity from the original [3] CP-ABE scheme, the local key management server (KMS) in the hospital that stores all users' attribute keys. The hospital outsources all encrypted documents to the third party cloud and outsources the private key generation operation to a semi-trusted third party key generation center (KGC). The KMS is responsible for most of the laborious key managerial tasks involving user attribute keys adding and deleting (revoke). There are four assumptions as the following: (1) Assume all servers are Semi-Trusted (Honest-but-curious); (2) Assume all users are registered to the cloud; (3) Assume cloud would not delegate (KEK) keys for unregistered users; and (4) Assume users could not collude with more than a semi-trusted party. We uses the entities of Figure 2 to illustrate the en-decryption operations.

C. *Construction*

Below are the constructions of DKUD-CP-ABE scheme.

(1) $Setup\ 1^{\lambda}) \rightarrow (PK, MK)$**.** The setup algorithm is executed by Key Management Server (KMS) during the Initial phase. It takes no input except implicit random exponents and random parameters. It outputs the public parameters PK as the Equation (1) which is published on cloud server and shared among all other entities and a master key MK stored secretly by KGC in Equation (2).

$$\text{Public Parameter, } PK = \left(G_0, g, h = g^{\beta}, e(g,g)^{\propto}\right) \tag{1}$$

$$\text{Master Key, } MK \quad (\beta, g^{\propto}) \tag{2}$$

(2) $Encrypt(PK, M, T, DEK) \rightarrow CT$**.** The encryption function takes in the public parameters PK, a message M, an access structure T, and the document encryption key DEK. The user first encrypts the document under the initial attribute keys. After that, the KMS will then perform attribute keying by selecting a set of attribute keys $< x: Key\left(att(x)\right) >$ and correspond to the ciphertext's

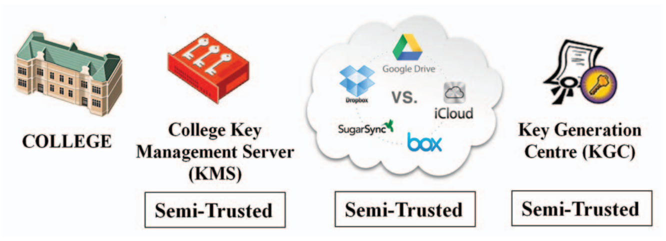


Fig. 2. DKUD Secure PHR Architecture Entities

attribute key on each leaf node x in T. The algorithm encrypts M and output a ciphertext CT as in Equation (3).

$$CT = \left(T, \tilde{C} = e(g,g)^{\propto s + ks}(M), C = h^{s}, \right. \tag{3}$$

$$\left. \forall y \in Y: C_y = g^{q_y(0)+\lambda_y}, C'_y = H\left(att(y)\right)^{q_y(0)+\lambda_y}\right). \tag{4}$$

(3) $\mathrm{KeyGen}(\mathrm{MK}, \mathrm{SA}) \rightarrow \left(\mathrm{PK}^{-1}{}_{\mathrm{u}}, \mathrm{PK}^{-1}{}_{\mathrm{attr_u}}\right)$ **.** The KeyGen algorithm is split into two parts. The first part is run by KGC that takes the master key MK and a set of attributes SA as input. It generates a random user identity and outputs an initial private key $\mathrm{PK}^{-1}{}_{\mathrm{u}}$ in Equation (4) which contains the static private key $\mathrm{PK}^{-1}{}_{\mathrm{static_u}}$, the Identity Key $\mathrm{K_{id_u}}$ as well as the initial attribute keys $\mathrm{PK}^{-1}{}_{\mathrm{init\_attr_u}}$ of the user. The second part is run by a key management center which performs attribute keying operation on the set of the initial attribute keys $\mathrm{PK}^{-1}{}_{\mathrm{init\_attr_u}}$ and output the Equation (5) keyed attributes $\mathrm{PK}^{-1}{}_{\mathrm{attr_u}}$.

(4) $\mathrm{Decrypt}(\mathrm{CT}, \mathrm{SK_{CT}}, \mathrm{KEK}) \rightarrow \mathrm{M}$ **.** The decrypt algorithm takes in three parameters, ciphertext CT which contains an access policy T, secret key for decryption $\mathrm{SK_{CT}}$ which contains a set of session-added attribute keys $\mathrm{PK}^{-1}{}_{\mathrm{attr\_t_u}}$ in Equation (6) and encryption key KEK which has the DEK encapsulated in it and requires valid user static key for utilization. If the DEK is successfully utilized and the attribute keys in $\mathrm{PK}^{-1}{}_{\mathrm{attr\_t_u}}$ satisfy the access policy T of CT, then the algorithm returns message, M as in Equation (7) otherwise, it returns $\perp$. For leaf node j, Decrypts each node x in T as

$$DecryptNode\ (CT, SK_{CT}, x) = \frac{e\left(D_j, C_x\right)}{e\left(D'_j, C'_x\right)} = \frac{e\left(D_j, C_x\right)}{e\left(D'_j, C'_x\right)}$$

$$= \frac{e\left(g^{id_u} \cdot H(j)^{ln_j+\lambda_j+t}, g^{q_x(0)+\lambda_x}\right)}{e\left(g^{ln_j+\lambda_j}, H\left(att(y)\right)^{q_x(0)+\lambda_x}\right)}$$

$$= \frac{\mathrm{e}\left(\mathrm{g}^{\mathrm{id_u q_x(0)+id_u\lambda_x}} \cdot \mathrm{g}^{\delta \mathrm{ln_j q_x(0)} + \delta \mathrm{ln_j}\lambda_\mathrm{x}} \cdot \mathrm{g}^{\delta\lambda_\mathrm{x} \mathrm{q_x(0)} + \delta\lambda_\mathrm{x}{}^2 + \mathrm{q_x(0)t} + \lambda_\mathrm{x}\mathrm{t}}\right)}{\mathrm{e}\left(\mathrm{g}^{\mathrm{ln_j}+\lambda_\mathrm{x}}, \mathrm{g}^{\delta \mathrm{q_x(0)} + \delta\lambda_\mathrm{x}}\right)}$$

$$= \frac{\mathrm{e}\left(\mathrm{g}^{\mathrm{id_u q_x(0)+id_u\lambda_x}} \cdot \mathrm{g}^{\delta \mathrm{ln_j q_x(0)} + \delta \mathrm{ln_j}\lambda_\mathrm{x}} \cdot \mathrm{g}^{\delta\lambda_\mathrm{x} \mathrm{q_x(0)} + \delta\lambda_\mathrm{x}{}^2 + \mathrm{q_x(0)t} + \lambda_\mathrm{x}\mathrm{t}}\right)}{\mathrm{e}\left(\mathrm{g}^{\mathrm{ln_j}\delta \mathrm{q_x(0)} + \mathrm{ln_j}\delta\lambda_\mathrm{x}} \cdot \mathrm{g}^{\lambda_\mathrm{x}\delta \mathrm{q_x(0)} + \delta\lambda_\mathrm{x}{}^2}, \mathrm{g}\right)}$$

$$= e(g,g)^{id_u q_x(0) + id_u\lambda_x + q_x(0)t + \lambda_x t}$$

$$= e(g,g)^{(q_x(0)+\lambda_x)id_u + (q_x(0)+\lambda_x)t} = e(g,g)^{(id_u+t)(q_x(0)+\lambda_x)} \tag{7}$$

Let $A = DecryptNode\ (CT, SK_{CT}, R)$
where R=root node= $e(g,g)^{(id_u+t)(q_R(0)+\lambda_x)}$
Let $(\mathrm{q_R}(0) + \lambda_\mathrm{x}) = \mathrm{s}$, $\mathrm{A} = \mathrm{e}(\mathrm{g}, \mathrm{g})^{(\mathrm{id_u}+\mathrm{t})\mathrm{s}}$

To decrypt a message:

$$\frac{\tilde{C}}{e(C,D\cdot DK)/_{A}} = \frac{\tilde{C}}{e\left(h^{s}, g^{\frac{\propto - id_u + k + 2i\ \ u + t}{\beta}}\right) / _{e(g,g)^{(id_u+t)s}}}$$

$$= \frac{[e(g,g)^{\propto s+ks}(M)]e(g,g)^{(id_u+\ t)s}}{e\left(g^{s\beta},\ \left(g^{\frac{\propto - id_u}{\beta}} \cdot g^{\frac{k+2id_u+t}{\beta}}\right)\right)}$$

$$= \frac{e(g,g)^{\propto s+ks} e(g,g)^{id_u s+ts}(M)}{e(g,g)^{\propto s - id_u s + ks + 2id_u s + ts}} = \frac{e(g,g)^{\propto s + id_u s + ks + ts}(M)}{e(g,g)^{\propto s + id_u s + ks +}} = M \quad (8)$$

D. DKUD-CP-ABE introduces two additional algorithms

(1) $\boldsymbol{KeyUpdate\left(H[\ \ _{id_u}], SEK_u, < SA_R >\right) \rightarrow \left(PK^{-1}{}_{attr\_t_u}, SEK_{\mathrm{KMS}}\right)}$. Key management Server run the KeyUpdate algorithm which takes two mandatory inputs, the hashed identity key of user and the session key of a login user, and an optional set of attribute(s) to be revoked $< SA_R >$. The algorithm outputs a set of updated session-added attribute keys $PK^{-1}{}_{attr\_t_u}$ as in Equation (8) and an updated shared session key as in Equation (9) between user, cloud and key KMS generated by KMS, SEK_{KMS}.

$$PK^{-1}{}_{attr_{t_u}} = \left(\forall j \in SA\colon D_j = g^{id_u} \cdot H(j)^{ln_j + \lambda_j + t}, D'_j = g^{ln_j + \lambda_j}\right) \quad (9)$$

$$SEK_{KMS} = \frac{(SEK_u + r)}{\beta} = t \in \mathbb{Z} \quad (10)$$

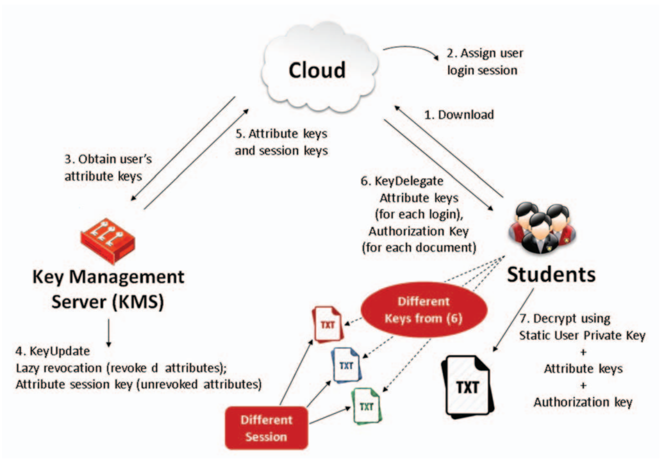
(2) $\boldsymbol{KeyDelegate(D\ \ , K_{id_u}, SEK_{\mathrm{KMS}}) \rightarrow KEK}$. The KeyDelegate algorithm takes as inputs the document delegation key DK, an identity key K_{id_u} and the session key shared between user, cloud and KMS. It checks and compare the identity key K_{id_u} againsts the revocation list and compares its registration date with the one of the delegation key DK. It returns a KEK as in Equation (10) with valid document encryption key DEK if the identity key K_{id_u} is registered before the document delegation key DK and the identity key K_{id_u} is not in the revocation list. Otherwise, it returns a KEK without DEK.

$$\text{either(authorize)}, g^{\frac{k+2id_u+t}{\beta}} \quad \text{or} \quad \text{(unauthorize)}, g^{\frac{2id_u+t}{\beta}} \quad (11)$$

Users logins to the system to obtain his attribute keys. The KMS performs lazy revocation(if any) and update his attribute keys to be usable under the particular login session. Then, he/she requests for the authorization key for the downloaded document. Upon obtaining the authorization key, they combines the keys that they obtained (attribute keys and authorization key) with his/her static user private keys for decrypting the document (Figure 3).

III. Security Analysis of DKUD

We now analyse the security of DKUD-CP-ABE which is being used in the scheme. The goal of the adversary is exploit the system and determine whether the given ciphertext is a proper DDH (Decisional Diffie-Hellman) tuples. A security



Cloud
2. Assign user login session
1. Download
5. Attribute keys and session keys
3. Obtain user's attribute keys
6. KeyDelegate Attribute keys (for each login), Authorization Key (for each document)
Key Management Server (KMS)
Students
Different Keys from (6)
4. KeyUpdate
Lazy revocation (revoke d attributes);
Attribute session key (unrevoked attributes)
7. Decrypt using Static User Private Key + Attribute keys + Authorization key
TXT
Different Session

Fig. 3. CP-ABE Key update and delegation

game similar to [1] is used to define a notion of Chosen Identity Key Security. Assuming the cloud would only delegate document keys for authorized users using their user identity keys. The security game demonstrates the DKUD-CP-ABE scheme is secure against the collusion attack between Key Generation Center and user under the Decisional Bilinear Diffie-Hellman assumption (DBDH). We proof that the security level of DKUD-CP-ABE against chosen identity key attack (CIKA) in the security game reduces to the hardness of the DBDH assumption.

(1) **Theorem 1.** If an adversary can break the DKUD-CP-ABE scheme in the CIKA model, then an algorithm can be constructed to play DBDH game with a non-negligible advantage ϵ.

(2) **Proof.** Suppose there exists a polynomial-time adversary A', that can attack the DKUD-CP-ABE scheme in the CIKA, we construct an algorithm $\mathcal{B}$ that can play the game with advantage $\frac{\epsilon}{2}$.

We let the challenger set the group G_0 and G_1 with an efficient bilinear map, e and generator g. The challenger flip a fair binary coin μ outside of $\mathcal{B}$'s view. If $\mu = 0$, the challenger sets $(lA, lB, lC, Z) = g^a, g^b, g^c, e(g,g)^{abc}$; otherwise it sets $(lA, lB, lC, Z) = g^a, g^b, g^c, e(g,g)^z$ from random $a, b, c, z \in Z_p$.

(3) **Init.** The algorithm $\mathcal{B}$ runs A'. A' selects an access structure T and chooses a set of attributes SA as $\{S_{1,\ldots,}S_{q'}\}$ which it wishes to be challenged upon.

(4) **Setup.** $\mathcal{B}$ sets the parameter $PK' = e(lA, lB) = e(g,g)^a$ of public key PK.$\mathcal{B}$ sends the public parameter PK to A'.

(5) **Phase 1.** A' adaptively makes private key request for an access structure T with a set of chosen attributes $S_{1,\ldots,}S_q$.

Case 1 - A is able obtain set of attributes that either do not satisfy the access structure T or *case 2* - the attributes set satisfy the access structure T but the identity key is invalid or revoked.To generate the private key, $\mathcal{B}$ assigns a polynomial q_x for every node in the access tree T, denoted by $SA = \{S_{1,\ldots,}S_q\}$. *Case 1*. $\mathcal{B}$ constructs the attribute keys for access tree T in which the root is satisfied. $\mathcal{B}$ chooses random $ln_j \in \mathrm{Z_p}$ for each leaf node j and bind them under A's identity. where ln_j is chosen randomly for each attribute. After that, $\mathcal{B}$ continuing defines the other remaining attribute keys. Upon completion of attribute key generation, $\mathcal{B}$ defines the static key of A'. The generated private key as in Equation (11, 12) is defined as $Attribute\ Keys,\quad {PK^{-1}}_{attr_{A'}}$

$$= \left(\forall j \in SA\colon\ D_j = g^{id_{A'}} \cdot H^{ln_j}, D_j' = g^{ln_j}\right) \tag{12}$$

$Static\ Private\ Key,\quad {PK^{-1}}_{id_{A'}}$

$$= \left(D = g^{(\propto - b - id_u)/\beta}, g^{\,b + id_u/\beta}\right) \tag{13}$$

Case 2. $\mathcal{B}$ constructs the attribute keys for access tree T in which the root is unsatisfied. $\mathcal{B}$ sets the attribute key for every leaf node j. For every unsatisfied leaf node j, $\mathcal{B}$ sets attribute keys using the same method as in *case 1* whereas for every other satisfied leaf node j' of T, it chooses random $ln_{j'} \in Z_p$ and bind them under A's identity. The structure of the private key for T is identical to that in DKUD-CP-ABE *KeyGen()*. $\mathcal{B}$ sets the attribute key to prevent uncover of the polynomial secret, $q_x(0)$ of node x in T if the correspondingd attribute key is satisfied, otherwise $\mathcal{B}$ would still knows only the invalid node x for T as x is unsatisfied. Therefore, $\mathcal{B}$ performs injection on the attribute key exponent as $(lB)g^{id_{A'}}$ for each attribute key j' in SA beforehand. The operations effectively render ${PK^{-1}}_{att_{A'}}$ in *case 1* for supporting $e(g,g)^{ab}$. Likewise, it sets exponent b on ${PK^{-1}}_{att_{A'}}$ in *case 2* to reduce to exponent a instead. Upon completion of attribute key generation, $\mathcal{B}$ define the static key of A'. The generated private key as in Equation (13, 14) is defined as

$Attribute\ Keys,\quad {PK^{-1}}_{attr_{A'}}$

$$= \left(\forall j' \in SA\colon\ D_{j'} = (lB)g^{id_{A'}} \cdot H^{ln_{j'}}, D_{j'}' = g^{ln_{j'}}\right) \tag{14}$$

$Static\ Private\ Key,\quad {PK^{-1}}_{id_{A'}}$

$$= \left(\mathrm{D} = \mathrm{g}^{(\propto + \mathrm{b} - \mathrm{id_u})/\beta}, \mathrm{g}^{\mathrm{id_u}/\beta}\right) \tag{15}$$

(6) **Challenge.** Adversary A' submits two identity keys K_{id_0}, K_{id_1} to the challenger $\mathcal{B}$. $\mathcal{B}$ flips a random coin $v \in \{0,1\}$ and delegate and sends the authorization key AK in Equation (16) under the K_{uid_v} to A' together with the encrypted document in Equation (15).

$$CT = \begin{pmatrix} T, \tilde{C} = m_v Z, C = h^s, \\ \forall y \in Y\colon\ C_y = g^{q_y(0)+\lambda_y},\ C_y' = H(att(y))^{q_y(0)+\lambda_y} \end{pmatrix} \tag{16}$$

$$\text{Authorization key } \mathrm{AK_{invalid}} = \mathrm{g}^{\frac{(\mathrm{k}+2\mathrm{id_u}+\mathrm{t})\mathrm{b}}{\mathrm{b}\beta}}, \mathrm{AK_{valid}}\mathrm{g}^{\frac{(\mathrm{k}+2\mathrm{id_u}+\mathrm{t})}{\beta}} \tag{17}$$

If $\mu\ =\ 0$ then $Z = e(g,g)^{abc}$.
$Let\ s = c, (PK')^s = ((e(g,g)^{ab})^c) = e(g,g)^{abc}$.
A' deduces that the ciphertext is a valid random encryption of message m_v. If $\mu = 1$, then $Z = e(g,g)^z$ which implies $\tilde{C} = m_v e(g,g)^z$ and A' is just given a random 4-tuple.

(7) **Phase 2.** Phase in is repeated with the same restriction with attribute set $S_{q+1,\cdots,}S_{q'}$.

(8) **Guess.** The adversary A' outputs a guess v' of v. If A' correctly guessed v' of v, the algorithm $\mathcal{B}$ outputs $\mu' = 0$ to indicate that it was given a valid DBDH-tuple otherwise, it will output $\mu' = 1$ as a result of just a random 4-tuple. The probability of A' in guessing v' when $\mu = 1$ is $\frac{1}{2}$ so $\mathcal{B}$ would guess $\mu' = 1$ with probability of $\frac{1}{2}$. On the other hand, if $\mu = 0$ then A' sees the encryption of m_v and could guess v with negligible advantage $\epsilon + \frac{1}{2}$ so does for $\mathcal{B}$. Finally, $\mathcal{B}$'s advantage as the Equation (17) in this game is

$$\frac{1}{2}Pr[\mu' = \mu|\mu = 0] + \frac{1}{2}Pr[\mu' = \mu|\mu = 1] - \frac{1}{2} = \frac{1}{2}\left(\frac{1}{2} + \epsilon\right) + \frac{1}{2}\frac{1}{2} - \frac{1}{2} = \frac{\epsilon}{2} \tag{18}$$

IV. Conclusion

According to the DKUD-CP-ABE scheme, the utilization of key delegations from an online cloud achieve direct revocations with backward and perfect forward secrecy through the binding of a session key for each private attribute key set. We realize the main functionality of the lookup attribute keys in KMS could be set as optional to provide perfect forward secrecy only where necessary, that is users could opt for "changing the attribute keys on each login" option only when using a publicly shared system to dispose of their sensitive key information upon logout. One of the weaknesses of such scheme is to maintain an always online server such as in [10] to perform required key delegations and updates even for direct revocations for the sake of instantaneity.

References

[1] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based Encryption For fine-Grained Access Control of Encrypted Data," Proceedings of the 13th ACM Conference on Computer and Communications Security, pages 89–98, 2006.

[2] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in Advances in Cryptology–Eurocrypt 2005, volume 3494, pages 457–473. Springer, 2005.

[3] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in Proceedings of the 2007 IEEE Symposium on Security and Privacy. IEEE Computer Society, 2007, pp. 321–334.

[4] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems," in Proceedings of the 13th ACM Conference on Computer and Communications Security, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 99–1.

[5] Z. Xu and K. Martin, "Dynamic user revocation and key refreshing for attribute-based encryption in cloud storage," in Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on, June 2012, pp. 844–849.

[6] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in In EUROCRYPT. Springer-Verlag, 1998, pp. 127–144.

[7] Minakshi V.Shinde, Prof.H.A.Hingoliwala, "Secure Cloud Storage using Multi Attribute Authority with Multi Central Authority," April 2015 Volume 3 Issue 4, International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC), ISSN: 2321-8169, PP: 1797 – 1801.

[8] Kan Yang, Xiaohua Jia, KuiRen,Bo Zhang,and Ruitao Xie, "DAC-MACS:Effective Data Access Control for Multiauthority Cloud Storage Systems," IEEE Transaction on Information Forensics and Security, Vol.8,No.11,Nov 2013.

[9] Lyes Touati and Yacine Challal, "Batch-Based CP-ABE with Attribute Revocation Mechanism for the Internet of Things".

[10] S. Jahid and N. Borisov., "PIRATTE: Proxy-based Immediate Revocation of ATTribute-based Encryption," 2012.