J Korea Industr Inf Syst Res Vol.19 No.1, 2014
http://dx.doi.org/10.9723/jksiis.2014.19.1.001

Secure Attribute-Based Access Control with a Ciphertext-Policy Attribute-Based Encryption Scheme†

Rifki Sadikin*, Young Ho Park**, and Kil Houm Park***

Abstract An access control system is needed to ensure only authorized users can access a sensitive resource. We propose a secure access control based on a fully secure and fine grained ciphertext-policy attribute-based encryption scheme. The access control for a sensitive resource is ensured by encrypting it with encryption algorithm from the CP-ABE scheme parameterized by an access control policy. Furthermore, the proposed access control supports non-monotone type access control policy. The ciphertext only can be recovered by users whose attributes satisfy the access control policy. We also implement and measure the performance of our proposed access control. The results of experiments show that our proposed secure access control is feasible.

Key Words : ciphertext-policy attribute-based encryption, public key encryption, access control

1. Introduction

Security service which ensures prevention of unauthorized use of a resource and limiting the flow of information is access control [1]. There are 3 types access control model known so far: discretional, mandatory and role-based. In discretional model the decision of access policy is created by a user by specifying the access mode of a resource. Discretional model is easy to implement and has flexibility for the user. However, the discretional model does not provide assurance on the flow of information in a system. In contrary, mandatory model assured the flow of information but in a rigid access control environment. While, the access structure of the role model combines the advantage of discretional and mandatory by partitioning the users into group of roles while role is defined as a set of actions and responsibilities associated with a working activity [2].

A new model for access control, called attribute-based access control (ABAC) is an access control model whereby the access policy use user's and resource's attribute altogether. The term of "attribute" has broader meaning than "role" in role-based access control. Attributes defines the property of a user or a resource. For example, a user's attributes in an academic information system can be $\{computerscience, cisl, phdstudent, 2009\}$ where $computerscience$ represents school's name,

† This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(NRF-2012R1A1A4A01002603)
* School of Electrical Engineering and Computer Science, Kyungpook National University, First Author
** Department of Electronics Engineering, Kyungpook National University, Corresponding Author (parkyh@knu.ac.kr)
*** School of Electronics Engineering, Kyungpook National University

$cisl$ represents laboratory's name, $phdstudent$ represents his/her role in the system and 2009 represents year of entrance. Then, an access policy is embedded to a resource to divide the set of attributes into two collections: authorized sets and unauthorized sets. For example, an access policy is represented as a Boolean function $(computerscience \wedge (professor \vee (phdstudent \wedge \neg cisl)))$. The user is granted an access only if his/her set of attributes satisfy the access policy.

Traditionally, the attribute-based access control mechanism employs a trusted server that records user's attributes and resource's protection in a trusted network [2]. However, this traditional approach solely depends on the security of the trusted server. Once the trusted server compromised, then the access control mechanism was also compromised. In general, an ABAC mechanism usually deploys a symmetric and a public cryptographic schemes to reduce the role of a trusted server [3,4]. However, any public cryptography schemes only support one-to-one relation between a secret key and a ciphertext which is not compliance with the nature of ABAC model. Moreover, such kind access control solutions still rely on a trusted server and a secure storage.

Ciphertext-policy attribute-based encryption (CP-ABE) is a class of attribute-based encryption system which improved the expressibility of access structure in the ciphertext [5]. CP-ABE systems can be used to provide access control in distributed environments since it can express complex relation between sensitive resources and principals. In CP-ABE, a plaintext is encrypted with an access structure where the access structure can be expressed in a Boolean function such as: ("student" AND "computer-science") OR "professor". A principal can recover the ciphertext, only if its attributes satisfy the access structure. In this case, a principal with a set of attributes {"student","electronics"} can not recover the ciphertext, while a principal with a set of attributes {"professor"} can recover the plaintext.

In this paper, we develop and evaluate an ABAC system built on a fully-secure CP-ABE with non-monotonic access structures. The CP-ABE is used for protecting the sensitive resource which encrypted with a certain access policy. Meanwhile, the access structure is used for authorizing access policy between resources and principals. Our CP-ABE scheme can express non-monotonic access policy which is problematic in CP-ABE schemes [5-9]. The non-monotonicity of our CP-ABE allows user to express the access policy with 4 logical gates: AND, OR, Threshold and NOT. Furthermore, our CP-ABE scheme is proven under fully security CP-ABE definition by applying dual encryption framework [10-11]. We also implement our ABAC system with pairing-based cryptography library [12]. We analyze the computation cost and the memory requirement of our ABAC system. We demonstrate that the cost of setup, key generation, encryption and decryption of our system are feasible.

2. Access Structure and CP-ABE

2.1 Access Structure

In a CP-ABE scheme, an access structure will be used in encryption algorithm to produce a ciphertext that parametrized by the given access structure. Formal definition of access structure is given as follows:

Definition 1. (Access Structure [13]). Let us define $\{P_1,...,P_n\}$ be a set of parties. A collection $\Lambda \in 2^{\{P_1,...,P_n\}}$ is monotone if $B \in \Lambda$ and $B \subseteq C$, implies $C \in \Lambda$. An access structure is a monotone collection Λ of non-empty subsets of $2^{\{P_1,...,P_n\}}$. The sets in Λ are called the authorized sets, and the sets not in Λ are called the unauthorized sets.

A non-monotonic access structure has more

expressibility than a monotonic access structure. CP-ABE schemes in [5-9] only allowed monotonic access structure. Whereas the recent CP-ABE can handle non-monotonic access structure [14-17]. Table 1 summarizes the differences between monotonic access structures and non-monotonic access structures.

<Table 1> Differences between monotonic and non-monotonic access structures

	Monotonic access structures	Non-monotonic access structures
Example	$\Lambda_1 = \{\{a_1,a_2\},\{a_1,a_3\},\{a_1,a_2,a_3\}\}$	$\Lambda_2 = \{\{a_1,a_2\},\{a_1,a_3\}\}$
Condition	if $B \in \Lambda$ and $B \subseteq C$, implies $C \in \Lambda$	No restriction
Boolean function	Monotonic function	Non-monotonic function
Boolean operators	AND($\wedge$) and OR($\vee$)	AND($\wedge$), OR($\vee$), NOT($\neg$)

2.2 Construction of the CP-ABE Scheme

We used a CP-ABE scheme which based on bilinear groups with composite order of 3 primes. We used variant of subgroup membership problems in [9] to prove that the CP-ABE construction is secure. Let us define a group generator:

$$(N = p_1 p_2 p_3, G, G_T, \hat{e}) \leftarrow gen(\lambda) \quad (1)$$

Where N is a composite number of 3 primes: p_1, p_2 and p_3, G and G_T are (additive/multiplicative) cyclic groups with order of N, and $\hat{e}: G \times G \rightarrow G_T$ is a bilinear map satisfying bilinearity and non-degenerate:

$$\exists\ g \in G,\ \forall\ a,b \in Z_N : \hat{e}(g^a, g^b) = \hat{e}(g,g)^{ab} \quad (2)$$

$$\exists\ g \in G : \hat{e}(g,g) \neq 1 \quad (3)$$

The bilinear groups with composite order enjoy orthogonallity between different subgroups $\hat{e}(g_i, g_j) = 1$ if $i \neq j$ where $g_i \in G_{p_i}, g_j \in G_{p_j}$ and G_{p_i} is sub group of G with order of p_i.

We present our CP-ABE construction as follows:

Setup Algorithm

$(pk, msk) \leftarrow Setup(\lambda, n)$, where λ is a security parameter and n is a bounded number for maximum size of the set of attributes, does the following:

(1) Generates bilinear groups
$(N = p_1 p_2 p_3, G, G_T, \hat{e}) \leftarrow gen(\lambda)$

(2) Selects group generator $g \leftarrow G_{p_1}$, $X_3 \leftarrow G_{p_3}$ and selects randomly $\alpha, a, v_1, v_2, \cdots, v_n \leftarrow Z_N$

(3) Selects a public cryptographic hash function $H(x) : Z_N \rightarrow G_{p_1}$

(4) Sets a public function
$V(x) = g^{a \triangle_{0,S}(x)} \prod_{i=1}^{n} (g^{v_i})^{\triangle_{i,S}(x)}$ where
$S = \{1,\ldots,n\}$ and $\triangle_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$
(known as Lagrange coefficient). $V(x)$ is actually a function that computes $g^{q(x)}$ where $q(x)$ is a random polynomial function degree of n, with $q(0) = a$.

(5) Sets public parameters
$pk = \{g, g^a, H(x), V(x), e(g,g)^{\alpha}\}$ and master keys $msk = \{\alpha, X_3\}$

Secret Key Generation Algorithm

$SK_X \leftarrow Keygen(pk, msk, X)$, where $X = \{x_1,\ldots,x_n\}$ is a set of attributes, does the following:

(1) Selects $t \leftarrow Z_N$ at random.

(2) Selects randomly
$R_3, R'_3, R_{3,1}, \ldots, R_{3,n}, R'_{3,1}, \ldots, R'_{3,n} \leftarrow Z_N$ by exponentiation of X_3.

(3) Sets the secret key as follows:

$$SK_X = \left\{ \begin{array}{c} X, K = g^{\alpha} g^{at} R_3, L = g^t R'_3 \\ \left\{ K_i = H(x_i)^t R_{3,i}, L_i = V(x_i)^t R'_{3,i} \right\}_{i \in \{1,\ldots,n\}} \end{array} \right\} \quad (4)$$

Encryption Algorithm

$CT_{\Lambda} \leftarrow Encrypt(pk, m, \Lambda)$, where $m \in G_T$ as a plaintext and Λ is an access structure related to a well-formed non-monotonic Boolean function $F(F_1, .., F_l)$ where F is a Boolean operator ($\wedge, \vee$ or $k-threshold$), and F_i is either a Boolean operator, a non-negated attribute y_i or negated attribute $\neg y_i$. Let us denote $Y = \{\hat{y_1}, .., \hat{y_l}\}$ as a set of attributes that defined the Boolean function $F(F_1, ..., F_l)$ where $\hat{y_i}$ is either a non-negated attribute ($\hat{y_i} = y_i$) or a negated attribute ($\hat{y_i} = \neg y_i$). The encryption algorithm does the following:

(1) Selects $s, r_1, ..., r_l \leftarrow Z_N$ at random.

(2) Runs the generalized secret sharing algorithm from [18] for the Boolean function $F(F_1, ..., F_l)$ and s recursively (from top to bottom) until F_i is either a non-negated attribute or negated attribute. Let us assume that $F(F_1, ..., F_l)$ and the secret share is s', we have 4 cases.:

(a) AND operator, $F = \wedge$:

① Selects $s_1, ..., s_{l-1} \leftarrow Z_N$ as shares for $F_1, ..., F_{l-1}$ respectively.

② Sets $s_l = \left(s' - \sum_{i=1}^{l-1} s_i \right) \bmod N$ as a share for F_l

(b) OR operator, $F = \vee$: sets s' as shares for $F_1, ..., F_l$

(c) Threshold operator, $F = k-threshold$:

① Selects $a_1, ..., a_{k-1} \leftarrow Z_N$ and set a polynomial function $p(x) = a_{k-1} x^{k-1} + ... + a_1 x + s'$

② Sets $s_i = p(i) \bmod N$ as a share for F_i

(d) A non-negated or negated attribute: sets share for the attribute $\hat{y_i}$ as $\lambda_i = s'$

(3) Sets the ciphertexts as follows:

$$CT_{\Lambda} = \left\{ \begin{array}{c} \Lambda, C_0 = m\, e(g,g)^{\alpha s}, C_1 = g^s, \left\{ C_{2,i} = g^{r_i} \right\}_{\forall\, i \in \{1,..,l\}} \\ \left\{ C_{3,i} = g^{a\lambda_i} H(y_i)^{r_i} \right\}_{\forall\, \hat{y_i} = y_i \in Y} \\ \left\{ C_{3,i} = g^{a(\lambda_i + r_i)}, D_{3,i} = V(y_i)^{r_i} \right\}_{\forall\, \hat{y_i} = \neg y_i \in Y} \end{array} \right\} \quad (5)$$

Decryption Algorithm

$m | \perp \leftarrow Decrypt(pk, SK_X, CT_{\Lambda})$, where SK_X is a secret key parameterized by a set of attributes X, CT_{Λ} is a ciphertext parametrized by an access structure, and Λ is an access structure related to a well-formed non-monotonic Boolean function $F(F_1, ..., F_l)$. The decryption does the following:

(1) Checks whether $X \in \Lambda$, if $X \notin \Lambda$ returns $\perp$

(2) If $X \in \Lambda$, then there is a set $\Gamma \subseteq Y$, $\Gamma = \{\hat{y_{i_1}}, .., \hat{y_{i_m}}\}$ where $m \le l$ satisfying Λ, we have two cases:

(a) If $\hat{y_{i_k}}$ is a non-negated attribute $\hat{y_{i_k}} = y_i$, then there must be $x_j \in X$ satisfying $y_i = x_j$, the decryption computes:

$$A_i = \frac{e(C_{3,i}, L)}{e(C_{2,i}, K_j)} = \frac{e(g^{a\lambda_i} H(y_i)^{r_i}, g^t R'_3)}{e(g^{r_i}, H(x_j)^t R_{3,j})} = e(g,g)^{at\lambda_i} \quad (6)$$

(b) If $\hat{y_{i_k}}$ is a negated attribute $\hat{y_{i_k}} = \neg y_i$, then $\forall\, x_j \in X$: $y_i = x_j$. Let $X' = X \cup y_i$, then the decryption computes:

$$A_i = \frac{e(C_{3,i}, L)}{e(C_{2,i}, \prod_{x_j \in X} L_j^{\Delta_{x_j, \hat{X}}(0)}) e(L, D_i)^{\Delta_{y_i, \hat{X}}(0)}}$$

$$= \frac{e(g,g)^{at\lambda_i} e(g,g)^{atr_i}}{e(g, \prod_{x_j \in \hat{X}} V(x_j)^{\Delta_{x_j, \hat{X}}(0)})^{tr_i}} = \frac{e(g,g)^{at\lambda_i} e(g,g)^{atr_i}}{e(g,g)^{tr_i \sum_{x_j \in \hat{X}} q(x_j) \Delta_{x_j, \hat{X}}(0)}}$$

$$= \frac{e(g,g)^{at\lambda_i} e(g,g)^{atr_i}}{e(g,g)^{q(0) tr_i}} = \frac{e(g,g)^{at\lambda_i} e(g,g)^{atr_i}}{e(g,g)^{atr_i}} = e(g,g)^{at\lambda_i} \quad (7)$$

(3) The decryption algorithm now can reconstruct the secret sharing in bottom-up manner until $F(F_1, ..., F_l)$ is satisfied, for each Boolean

operator F_i we have 3 cases:

(a) AND operator: $\wedge(F_1,...,F_l)$ and shares are $\{\lambda_1,...,\lambda_l\}$, the decryption computes:

$$A_i = \prod_{i=1}^{l} e(g,g)^{at\lambda_i} = e(g,g)^{s_i} \quad (8)$$

(b) OR operator: $\vee(F_1,...,F_l)$ and shares are $\{\lambda_1,...,\lambda_l\}$, the decryption algorithm chooses λ_k where F_k is satisfied and yields:

$$A_i = \prod_{i=1}^{l} e(g,g)^{at\lambda_i} = e(g,g)^{s_i}$$
$$A_i = e(g,g)^{at\lambda_k} = e(g,g)^{s_i} \quad (9)$$

(c) Threshold operator:
$k-threshold(F_1,...,F_l)$ and shares are $\{\lambda_1,...,\lambda_l\}$, chooses $J \subseteq \{1,..,l\}$ where $|J| = k$ and $\forall\, j \in J$, F_j is satisfied. The decryption algorithm computes:

$$A_i = \prod_{\forall\, j \in J} e(g,g)^{at\lambda_j \triangle_{j,J}(0)} = e(g,g)^{at\, s_i} \quad (10)$$

(4) At the end of reconstruction of the secret share the decryption algorithm yields $A_{root} = e(g,g)^{ats}$ and recover the plaintext as follows:

$$C_0 \frac{A_{root}}{e(C_1.K)} = m\, e(g,g)^{\alpha s} \frac{e(g,g)^{ats}}{e(g^s, g^{\alpha} g^{at})} \quad (11)$$
$$= m\, e(g,g)^{\alpha s} \frac{e(g,g)^{ats}}{e(g,g)^{\alpha s} e(g,g)^{ats}} = m$$

Correctness of the CP-ABE scheme

The decryption algorithm shows that our CP-ABE scheme satisfied *correctness* requirement.

2.3 Security of the CP-ABE Scheme

The proposed CP-ABE scheme is proven to be fully secure though a model of security based on indistinguishable game under chosen plaintext (ciphertext) attack (CPABE-IND-CPA). The security proof used dual encryption system from [10]. We introduce semi-functional ciphertext and secret key by using elements from G_{p_2}. First, we define the $GAME_{REAL}$ as follows:

- **Setup**. The challenger runs the $setup$ algorithm and gives the public parameters pk to the adversary.
- **Phase 1**. The adversary makes repeated private keys corresponding to arbitrary sets of attributes $X_1,...,X_{q_1}$.
- **Challenge**. The adversary submits two equals length messages m_0 and m_1. In addition, the adversary gives a challenge non-monotone access structure Λ^* given that none of the sets $X_1,...,X_{q_1}$ from Phase 1 satisfy the challenge access structure Λ^*. The challenger flips a random coin $b \leftarrow \{0,1\}$ and encrypt m_b under Λ^*.
- **Phase 2**. Phase 1 is repeated with the restriction that none of sets of attributes $X_{q_1},...,X_q$ satisfy the access structure corresponding to the challenge access structure Λ^*.
- **Guess**. The adversary outputs a guess b' of b.

The advantage of an adversary A in this game is defined as

$$Adv_{\Gamma,A}^{CPABE-IND-CPA} = \left| \Pr[b = b'] - \frac{1}{2} \right| \quad (12)$$

This model can be extended to handle chosen-chipertext attacks by allowing for decryption queries in Phase 1 and Phase 2.

Definition 3. (Secure CP-ABE Scheme [17]). A CP-ABE scheme Γ is secure if all probabilistic polynomial time (PPT) adversary A has at most

a negligible advantage in CPABE-IND-CPA game.

We also define $GAME_{FINAL}$ where the challenger always return a semi-functional secret key in secret key query (phase 1 and phase 2) and return a semi-functional ciphertext with random element in challenge phase. Similar to [8], we can show that $GAME_{REAL}$ is indistinguishable with $GAME_{FINAL}$ for any PPT adversary to satisfy Definition 3.

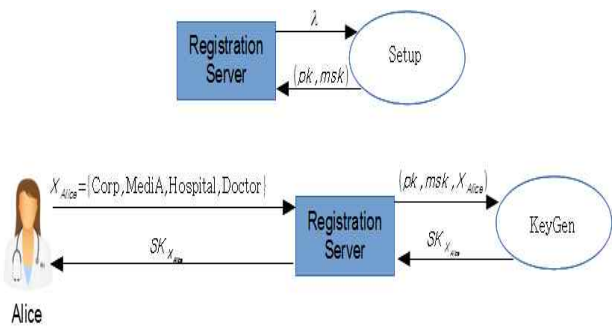
3. Attribute-Based Access Control System

3.1 Set of Attributes and Secret Key

In ABAC system, a user is not identified by an identity but a set of attributes which is a subset over an attribute space $X \subseteq A$ where $A = \{a_1, \ldots, a_x\}$. Every user who wants to be a decryptor must generate a secret key parameterized by his/her attributes set SK_X. For example, in a simplified *health insurance* system from [19] the attributes space is:

$$U = \begin{Bmatrix} Corp, CorpA, CorpB, Medi, MediA, \\ Hospital, HospitalA, HospitalB, Phar, PharA, \\ Doctor, Nurse, Pharmacist, BillingPersonel \end{Bmatrix} \quad (13)$$

Let assume there is a doctor named Alice which works for hospital $HospitalA$ and covers insurance issued for a company $Corp$ under $MediA$ prescription plan. Therefore, a set of attributes that parameterized secret key for Alice is $X_{Alice} = \{Corp, MediA, Hospital, Doctor\}$. In order to use the *health insurance* system, she must generate a secret key parameterized by X_{Alice} as illustrated in Figure 1.



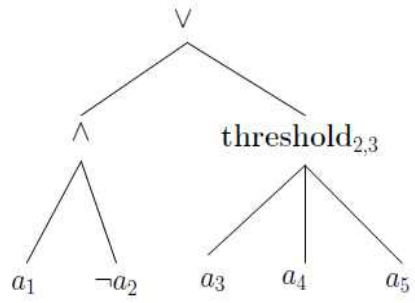
<Fig. 1> Registration server setup and principal secret key generation.

3.2 Access Policy and Ciphertext

Let us define $O = \{o_1, \ldots, o_y\}$ as a set of sensitive objects in an access control system. Access control for an object o_i is defined by a non-monotonic access structure Λ. The non-monotonic access structure Λ is related to a non-monotonic Boolean function $F(a_1, \ldots, a_l)$. The function can be transformed to a monotonic Boolean function $F(\widehat{a_1}, \ldots, \widehat{a_l})$ containing AND, OR and k-threshold functions. $\widehat{a_1}$ is either a non negated attribute $\widehat{a_k} = a_k$ or a negated attribute $\widehat{a_k} = \neg a_k$.

One of examples of non-monotonic Boolean functions is
$F(a_1, \ldots, a_l) = \vee(\wedge(a_1, \neg a_2), 2-threshold(a_3, a_4, a_5))$.

The function $F(a_1, \ldots, a_l)$ can be represented as a tree structure as Figure 2. Note that our system allowed NOT gate in leaf nodes. However, this limitation does not affect the expressibility of the access structure.



<Fig. 2> An example of access control.

The access control system employs encryption algorithm from the CP-ABE scheme to protect a

<Table 2> Comparisons between our CP-ABE and other CP-ABE Schemes

	Beth [6]	Cheung [15]	Lewko [8]	Okamoto [16]	Ours
Security	Full	Selective	Full	Full	Full
Access structure	Monotone	Non-monotone	Monotone	Non-monotone	Non-monotone
$\|pk\|$	$O(1)$	$O(\|U\|)$	$O(\|U\|)$	$O(\|U\|)$	$O(n)$
$\|SK_X\|$	$(2\|X\|+1)\|G\|$	$(2\|U\|+1)\|G\|$	$(\|X\|+3)\|G\|$	$(4\|X\|+3)\|G\|$	$(2\|X\|+2)\|G\|$
$\|CT_\Lambda\|$ $\Lambda : F(y_1,..,y_l)$	$(2l+1)\|G\|$ $+\|G_T\|$	$(\|U\|+1)\|G\|$ $+\|G_T\|$	$(2l+1)\|G\|$ $+\|G_T\|$	$(3l+9)\|G\|$ $+\|G_T\|$	$(2l+1)\|G\|$ $+\|G_T\|$
Number of pairing (m matched)	$2m$	$\|U\|$	$2m+1$	$7m+2$	$2m+1 \leq$ $\leq 3m+1$

sensitive object o_k. The encryption algorithm requires an access structure Λ, our system used a non-monotonic Boolean function $F(\hat{a_1},...,\hat{a_l})$ to segregate users into authorized users or unauthorized users of the object o_k. Principally, when an sensitive object o_k want to be distributed across unsecured network the encryption algorithm $CT_\Lambda \leftarrow encrypt(pk, K, \Lambda : F(\hat{a_1},...,\hat{a_l}))$ is invoked and the encryptor distributes $E_K(o_k), CT_\Lambda$ instead of o_k where $E_K(.)$ is symmetric block cipher with a key K.

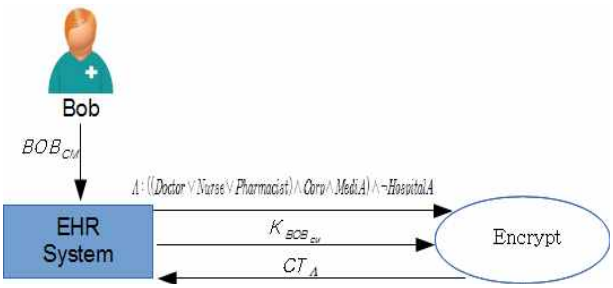
In the same example as in previous section, a patient in a simplified *health insurance* system could have electronic health record (EHR) that contains current medication history of the patient Bob. Suppose only doctors, nurses, and pharmacists that support insurance for any corporation $Corp$ under $MediA$ prescription plan and are not working for $HospitalA$ can access his current medication. Let us denote Bob_{CM} represents the current medication of Bob. When Bob creates Bob_{CM} through a EHR system, the EHR system supplies an access policy presented in a non-monotonic Boolean function:

$$\Lambda : ((Doctor \vee Nurse \vee Pharmacist) \wedge Corp \wedge MediA) \wedge \neg HospitalA \quad (14)$$

Just like in key generation, the EHR system employs the same hash function $H_{attr} : \{0,1\}^n \rightarrow Z_N$. It selects random $K \leftarrow \{0,1\}^N$ as a secret key for a symmetric encryption algorithm $E_K(.)$. Then, it calls:

$$CT_{\Lambda_{Bob}} \leftarrow encrypt(pk, K, \Lambda : (Doctor \vee Nrse \vee Pharmacist) \wedge Corp \wedge MediA) \wedge \neg HospitalA), E_K(Bob_{CM}) \quad (15)$$

At the end, EHR publishes/stores $E_K(Bob_{CM}), CT_{\Lambda_{Bob}}$ as illustrated in Figure 3.

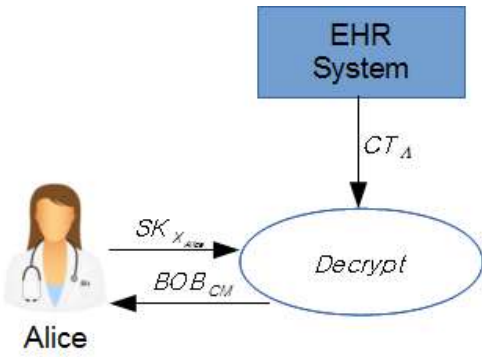


<Fig. 3> Encryption of a resource with an access policy.

3.3 Accessing Object

When a user wants to access a protected object o_k in a ciphertext CT_Λ, the user must provide a secret key SK_X parameterized by a set of

attributes X. If $X \in \Lambda$ (X is an authorized set according to the access structure Λ) then the user can decrypt CT_{Λ} and receive the original plaintext m. For example, for a ciphertext $CT_{\Lambda'}$ where the access structure is defined by a Boolean function: $\Lambda' : F(a_1,...,a_l) = \vee(\wedge(a_1,\neg a_2),2-threshold(a_3,a_4,a_5))$. The user with a secret key SK_X parameterized by $X = \{a_1,a_3,a_4,a_5\}$ can decrypt $CT_{\Lambda'}$. While, $X = \{a_1,a_2,a_3,a_5\}$ can not decrypt $CT_{\Lambda'}$.



<Fig. 4> Access of a sensitive resource by a principal.

In the simplified *health insurance* system example as illustrated in Figure 4, Alice who has $SK_{X_{Alice}}$ where $X_{Alice} = \{CorpA, MediA, Hospital, Doctor\}$ can retrieve a protected object of Bob's recent medication (Bob_{CM}) stored by the EHR system since X_{Alice} satisfied the access policy of Bob_{CM}

$$\Lambda : ((Doctor \vee Nurse \vee Pharmacist) . \\ \wedge Corp \wedge MediA) \wedge \neg HospitalA$$

To recover Bob_{CM}, Alice invokes:

$$K = decrypt(pk, CT_{\Lambda_{Bob}}, SK_{X_{Alice}}), Dec_K(Enc_K(Bob_{CM})) \quad (16)$$

4. Performance

4.1 Comparison

We compare our CP-ABE construction with other existing schemes in regards of monotonicity of the access structure, security of CP-ABE, space requirement (public parameters, secret key and ciphertext length) and computation cost (number of pairing pin decryption). From Table 2, our CP-ABE scheme achieves full security CP-ABE definition and non-monotone access structure. Most of the existing schemes do not achieve these properties. The size of public parameters pk of ours is in linear function of n where n is maximum size of a set of attributes to parameterize a secret key. While, others are in linear function of $|U|$ (attribute space size). Therefore, our CP-ABE scheme supports large universe construction.

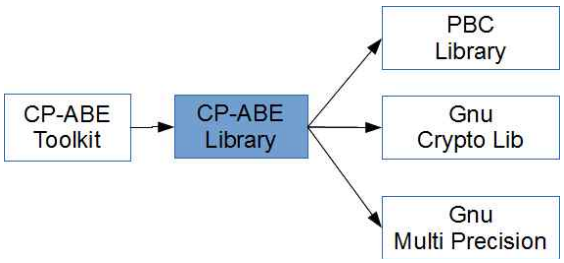
The size of a secret key and a cipher text in our CP-ABE depends on the choice of bilinear groups. We denote $|G|$ and $|G_T|$ to represent the size of element in group G and G_T respectively in bits. The size of a secret key of our CP-ABE $|SK_X|$ is in linear function of $|X|$ (which is the same as others CP-ABE schemes except Cheung [15]). The exact size of a secret key in our CP-ABE scheme is $(2|X|+2)|G|$. While, the size of a ciphertext of our CP-ABE $|CT_{\Lambda}|, \Lambda : F(y_1,..,y_l)$ is in linear function of l (the number of attributes in the access structure Λ) where the exact size of a ciphertext in our CP-ABE scheme is $(2l+1)|G|+|G_T|$.

4.2 Implementation

We implement our proposed access control system in C language and compiled the implementation with GNU C compiler. The program structure of our implementation is given by Figure 5. We use three libraries: pairing based cryptography (PBC) library, GNU multi-precision and GNU Crypto library. PBC library is developed by Ben Lynn of Standford University as a part of his PhD thesis [12]. PBC library provides operation in bilinear groups and pairing function. GNU Crypto

library provides standard cryptographic systems which includes cryptographic hash functions and key encapsulation methods [20]. While, GNU multi-precision library provides functions to compute big number arithmetic [21]. The CPABE toolkit from [6] provides interface for using our CP-ABE library.

The CP-ABE library contains supporting and main functions. The supporting functions include a group generator, memory (de)allocators and cryptographic hashes. While, the main provides the implementation of 4 algorithm in CP-ABE scheme: $setup$, $keygen$, $encrypt$, and $decrypt$.



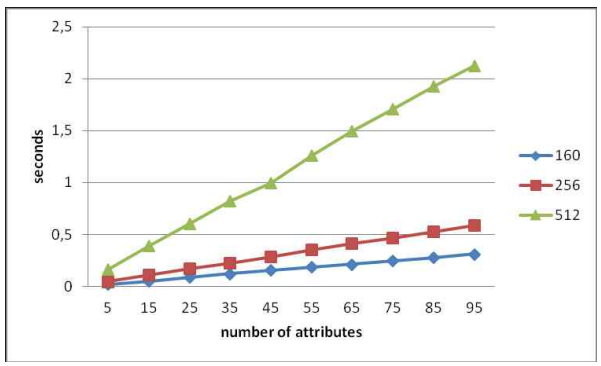
<Fig. 5> Program structure of access control based on CP-ABE.

4.3 Experiment Results

We do some experiment to get actual time needed for the access control system executing 4 basic functions: $setup$, $keygen$, $encrypt$, and $decrypt$. We vary the number of attributes and security parameter λ (size of bits). The experiment is conducted in a 64-bit dual core processor each with speed 3.30GHz and 3.70 GHz with 8GByte RAM.

We conducted the experiment for 3 different security parameters: 160. 256 and 512 bits. For each security parameters we execute $setup$, $keygen$, $encrypt$, and $decrypt$ for different size of attributes. In $setup$, number of attributes represents the size of attribute space. While for $keygen$ and $encrypt$, it represents the size of attributes set in $keygen$ and the number of attributes in the access structure.

Figure 6 shows the Setup running time consumption is linear function in the size of attribute space. Typically, the size of attribute space is large. However, the Setup function is only executed once throughout the life time of the access control system. When $\lambda = 160$ (the size of element in group G and G_T are 160 bits) we found the relation between the running time and the size of attributes is $T_{Setup,160} = 0,00319\,n + 0.00796$ in seconds by linear regression. We also find the relation for other security parameters are $T_{Setup,256} = 0,00598\,n + 0.0180$ and $T_{Setup,512} = 0,02195\,n + 0.05042$. Surely there is trade-off between the size of security parameter and time consumption, but from our measurement when $\lambda = 160$ and $\lambda = 256$, the time consumption for Setup function still in order of unit of seconds for n in order of thousands and for $\lambda = 512$ is in the order hundreds.

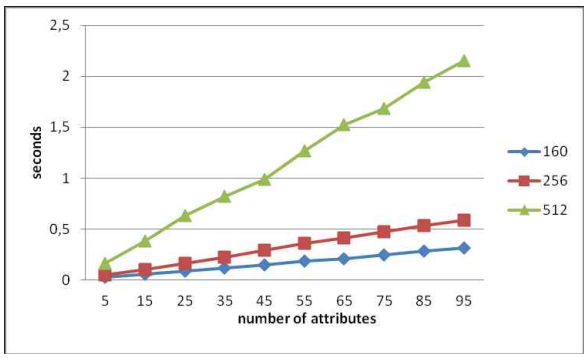


<Fig. 6> Setup time consumption.

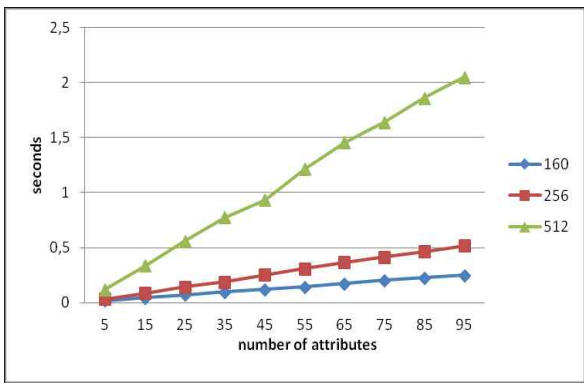
Figure 7 illustrates the cost of Secret Key Generation function. The number of attributes is the size of a set of attributes $|X|$ to parameterize secret key. Our experiments show that the cost of secret key generation is a linear function of $|X|$ where $T_{KeyGen,160} = 0,00322|X| + 0.00742$, $T_{KeyGen,256} = 0,00608|X| + 0.01481$ and $T_{KeyGen,512} = 0,02205|X| + 0.05423$ in seconds.

We found similar result for the cost of encryption. Figure 8 shows the cost of encryption function is in linear function of l(the number of attributes in the access structure Λ) where $T_{Encryption,160} = 0,00260\,l + 0.00426$, $T_{Encryption,256} = 0,00546\,|X| + 0.00449$ and $T_{KeyGen,512} = 0,02168\,|X| + 0.01071$ in seconds.

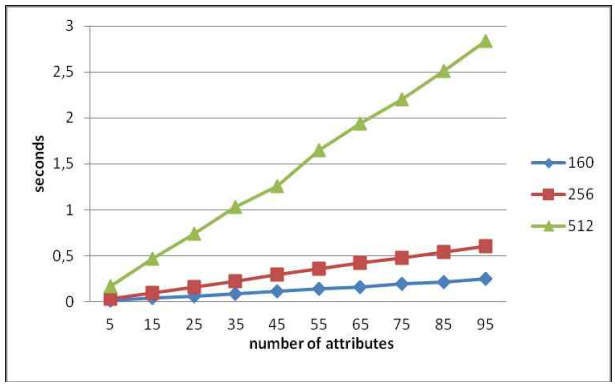
We found that the cost of decryption function is costlier than other functions as illustrated in Figure 9. This is because the decryption function uses pairing function from bilinear groups. Pairing function is the costliest than other group operations in bilinear groups. The cost of decryption function is in linear of m, where m is the number of matching attributes between the set attributes X and attributes in the access structure Λ, are



<Fig. 7> Key generation time consumption.



<Fig. 8> Encryption time consumption.



<Fig. 9> Decryption time consumption.

$T_{Decryption,160} = 0,00258m + 0.0003$,
$T_{Decryption,256} = 0,00635\,m + 0.00387$ and
$T_{Decryption,512} = 0,02955\,|X| + 0.00242$ in seconds.

This shows the cost of key generation, encryption or decryption when λ is 160, 256, and 512.

5. Conclusion

This paper proposed an access control system based on a CP-ABE scheme. We used a fully secure CP-ABE constructions that support non-monotonic access control. The proposed access control allows complex relation between principals and resources. Principals give the set of its attributes for generating a secret key parameterized with its attributes. While, resources are given an access policy in a non-monotonic Boolean function to allow/disallow access based on principal's attributes. The performance of our access control system depends on the performance of the proposed CP-ABE. From analysis we found that our proposed CP-ABE retains fully secure and non-monotonicity of access structure and has reliable cost regarding memory requirement and computation cost compared to other existing CP-ABE schemes. The experiments shows that the computation cost of our access control. When the security parameter is 160 or 256, the order of the

number of attributes is thousands. However, when the security parameter is 512, the number of attributes is in the order hundreds.

References

[1] S. Mazgon (Editor), "Approved ITU-T Security Definition", *Security Compendium*, ITU-T, 2005

[2] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, C. E. Youman, "Role-Based Access Control Models", *Computer*, vol. 29, no. 2, pp. 38-47, 1996.

[3] Keith Frikken, Mikhail Atallah, Jiangtao Li, "Attribute-Based Access Control with Hidden Policies and Hidden Credentials", *IEEE Transaction on Computers*, vol. 55, no. 10, pp. 1259-1270, 2006.

[4] Rifki Sadikin, YoungHo Park, KilHoum Park, and SangJae Moon "Universal Composability Notion for Functional Encryption Schemes," Journal of the Korea Society of Industrial Information Systems, vol.18, no.3, pp.17-26, 2013.

[5] Amit Sahai, and Brent Waters. "Fuzzy identity-based encryption." *Advances in Cryptology - EUROCRYPT 2005*. pp. 457-473, 2005.

[6] J. Bethencourt, Amit Sahai, and Brent Waters. "Ciphertext-policy attribute-based encryption." *Security and Privacy,* IEEE Symposium, 2007.

[7] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Proceedings of the 35th international colloquium on Automata, Languages and Programming*, Part II, ICALP '08, pp. 579 - 591, 2008.

[8] Allison Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption.", *Advances in Cryptology - EUROCRYPT 2010*. pp. 62-91, 2010.

[9] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in Proceedings of the 14th international conference on Practice and theory in public key cryptography conference on Public key cryptography, pp. 53 - 70, 2011.

[10] B. Waters. "Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions." *Advances in Cryptology-CRYPTO 2009*. pp. 619-626, 2009.

[11] Allison Lewko and Brent Waters. "New techniques for dual system encryption and fully secure HIBE with short ciphertexts." *Theory of Cryptography*. Springer Berlin Heidelberg, pp. 455-479, 2010.

[12] Ben Lynn, *On the Implementation of Pairing-Based Cryptography*, Phd Thesis, Department of Computer Science, Stanford University, USA, 2007.

[13] A. Beimel, Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, 1996.

[14] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker, "Efficient and provable secure ciphertext-policy attribute-based encryption schemes," in *Proceedings of the 5th International Conference on Information Security Practice and Experience*, pp. 1 - 12, 2009.

[15] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 456 - 465, 2007.

[16] T. Okamoto and K. Takashima, "Fully secure functional encryption with general relations from the decisional linear assumption," in Advances in Cryptology CRYPTO 2010, pp. 191 - 208, 2010.

[17] Dan Boneh, Amit Sahai, and Brent Waters. "Functional encryption: Definitions and challenges." *Theory of Cryptography*. Springer Berlin Heidelberg, pp. 253-273, 2011.

[18] J. Benaloh and J. Leichter, "Generalized secret sharing and monotone functions," in

Proceedings on Advances in cryptology, CRYPTO '88, pp. 27 - 35, 1990.
[19] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems.", *Proceedings of the 13th ACM conference on Computer and communications security*, pp. 99-112, 2006.
[20] C. Marshall and R S Naffah, *Programming with GNU Crypto*, The Free Software Foundation, 2003.
[21] T. Granlund, *GNU MP: The GNU Multiple Precision Arithmetic Library*, The Free Software Foundation, 2013.

Rifki Sadikin received his B.S. in electrical engineering from Gadjah Mada University, Yogyakarta, Indonesia in 1999, M.S. degree in computer science from Indonesia University, Jakarta, Indonesia in 2004. From 2009 until now, he is a PhD student in School of Electrical Engineering and Computer Science at Kyungpook National University, Korea. His research interests include information security, computer network, and distributed system.

Young Ho Park received his BS, MS, and Ph. D degrees in electronic engineering from Kyungpook National University, Daegu, Korea in 1989, 1991, and 1995, respectively. He is currently a professor in the Department of Electronics Engineering at Kyungpook National University. In 1996-2008, he was a professor in the School of Electronics and Electrical Engineering at Sangju National University, Korea. In 2003-2004, he was a visiting scholar in the School of Electrical Engineering and Computer Science at Oregon State University, USA. His research interests include computer networks and information security.

Kil Houm Park received his B.S. degree from Kyungpook National University, Daegu, Korea, in 1982 and his M.S. and Ph.D. degrees in electronic engineering from KAIST, Daejeon, Korea, in 1984 and 1990. He has been a Professor with the School of Electronics Engineering, Kyungpook National University, since 1984. His current research interests include medical image processing, fingerprint recognition, computer vision, and information security.

논 문 접 수 일 : 2013년 07월 02일
1차수정완료일 : 2013년 09월 16일
2차수정완료일 : 2013년 12월 24일
게 재 확 정 일 : 2014년 01월 16일