Naruse et al. Human-centric Computing and Information Sciences (2015) 5:8
DOI 10.1186/s13673-015-0027-0

Human-centric Computing and Information Sciences
a SpringerOpen Journal

RESEARCH Open Access

Provably secure attribute-based encryption with attribute revocation and grant function using proxy re-encryption and attribute key for updating

Takeru Naruse[1*], Masami Mohri[2†] and Yoshiaki Shiraishi[3†]

* Correspondence:
naruse.takeru@nitzlab.com
†Equal contributors
[1]Nagoya Institute of Technology, Nagoya, Japan
Full list of author information is available at the end of the article

Abstract

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) is suitable for data access control on a cloud storage system. In CP-ABE, the data owner encrypts data under the access structure over attributes and a set of attributes assigned to users is embedded in user's secret key. A user is able to decrypt if his attributes satisfy the ciphertext's access structure. In CP-ABE, processes of user's attribute revocation and grant are concentrated on the authority and the data owner. In this paper, we propose a ciphertext-policy attribute-based encryption scheme delegating attribute revocation processes to Cloud Server by proxy re-encryption. The proposed scheme does not require generations of new secret key when granting attributes to a user and supports any Linear Secret Sharing Schemes (LSSS) access structure. We prove that the proposed scheme is secure against attack by unauthorized users and Cloud Server.

Keywords: Cryptographic cloud storage; Ciphertext-policy attribute-based encryption; Attribute revocation and grant; Proxy re-encryption

Background

Sharing of data on a cloud storage has a risk of information leakage caused by service provider's abuse. In order to protect data, the data owner encrypts data shared on the cloud storage so that only authorized users can decrypt.

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [1,2] is suitable for data access control in the cloud storage system. The authority manages the attributes in the system. The data owner chooses an access structure and encrypts message under the access structure. The set of attributes assigned to users is embedded in his secret key. A user is able to decrypt a ciphertext if his attributes satisfy the ciphertext's access structure.

There are user's attribute revocation and grant in CP-ABE. In simple processes of user's attribute revocation, when his attributes are revoked, the data owner re-encrypts the shared data so that revoked user cannot decrypt. Then, the authority redistributes new secret keys so that other users can decrypt. In simple processes of user's attribute grant, the authority generates a new secret key. These simple processes are concentrated on the data owner and the authority.

Some attribute revocable CP-ABE schemes have been proposed [3-5]. Yu et al. [3] proposed a scheme combining CP-ABE with proxy re-encryption. The authority can delegate re-encryption and secret key update to proxy servers. However, this scheme has a limitation in access policy because it can only express "AND" policy. Hur et al. [4] proposed a scheme using key encryption keys (KEKs). A service provider distributes KEKs to each user. The service provider re-encrypts a ciphertext by an attribute group key. Then, he encrypts attribute group key by using KEKs so that authorized user can decrypt. As the number of system users has increases, the number of KEKs also increases and management becomes complicated. Liang et al. [5] proposed a scheme using user information (UI). UI is generated by Revocation Tree and Revocation List. An authorized user can decrypt ciphertexts by using secret key and UI. In this scheme, users whose attributes are revoked lose the access rights to all shared data by attribute revocation processes.

Moreover, in these schemes [3-5], the authority needs to generate a new key when granting attribute to users.

In this paper, we propose a CP-ABE scheme delegating attribute revocation processes to Cloud Server by proxy re-encryption and meets the following requirements.

1) *Support any Linear Secret Sharing Schemes (LSSS) access structure.* In existing schemes, there are following three expression methods of the access structure: tree, vector and matrix.
   The schemes using tree (for example the schemes of [1,4]) are not given the security proof under the standard model because they use hash functions for encryption.
   The schemes using vector whose elements are combined by AND condition are given the security proof under the standard model (for example the scheme of [3]) but they cannot perform flexible access control because only "AND" can be used for decryption conditions.
   By using matrix, for example a LSSS matrix, it is possible to use "AND" and "OR" for decryption conditions, that is, it is possible to perform a fine-grained access control and give the security proof under the standard model [2,5]. However, the revocation function is not shown in [2], and the scheme of [5] can revoke the only specified users but cannot revoke the only specified attributes.
2) *Revoke the only specified attribute (attribute level user revocation).* In the scheme of [5], users whose attributes are revoked lose the access rights to all shared data. To perform a fine-grained access control of shared data, it is desirable to revoke the only specified attributes. The schemes of [3,4] meet this requirement.
3) *Does not require the generation of new secret key when granting attribute to user.* In the way that the authority generates a new secret key and sends the key to a user every time it grants attributes to a user, the calculation amount of the authority is large in the case attributes are frequently granted. To avoid focusing the process of user's attribute grant in the authority, we enable cloud servers to update user's secret key. As far as we know, there is no scheme that meets this requirement.

We prove that the proposed scheme is secure under the standard model. We define attack model 1 as attack by unauthorized users and attack model 2 as attack by Cloud Server. We prove the proposed scheme is IND-CPA secure in the standard model against each attack model.

Methods

Preliminaries

Bilinear Maps

Let G_1, G_2 be two cyclic groups of prime order p. Let P be a generator of G_1. A bilinear map is a map $e : G_1 \times G_1 \rightarrow G_2$ with the following properties:

1) Bilinearity: for all $P, Q \in G_1$ and $a, b \in Z_p$, we have $e(aP, bQ) \rightarrow e(P, Q)^{ab}$.
2) Non-degeneracy: $e(P, P) \neq 1$.
3) Computability: There is an efficient algorithm tocompute $e(P, Q)$ for all $P, Q \in G_1$.

Linear Secret Sharing Scheme (LSSS)

Definition 1 (Linear Secret Sharing Schemes (LSSS) [2,6]) A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over Z_p) if

1) The shares for each party form a vector over Z_p.
2) There exists a matrix an M with l rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots, l$, te i'th row of M we let the function ρ defined the party labeling row i as $\rho(i)$. When we consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in Z_p$ is the secret to be shared, and $r_2, \ldots, r_n \in Z_p$ are randomly chosen, then Mv is the vector of l shares of the secret s according to Π. The share $(Mv)_i$ belongs to party $\rho(i)$.

Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2, \ldots, l\}$. Then, there exist constants $\{\omega_i \in Z_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. Futhermore, there these constants $\{\omega_i\}$ can be found in time polynomial in the size of the share-generating matrix M [6].

Decisional Parallel Bilinear Diffie-Hellman Exponent Assumption

Choose a group G_1 of prime order p according to the security parameter. Let a, s, $b_1, \ldots, b_q \in Z_p$ be chosen at random and $P \in G_1$ be a generator of G_1. If an adversary is given $\vec{y} =$

$$P, sP, aP, \ldots, a^q P, \quad , a^{q+2} P, \ldots, a^{2q} P$$

$$\forall_{1 \leq j \leq q} \; s \cdot b_j P, \left(a / b_j\right) P, \ldots, \left(a^q / b_j\right) P, \quad , \left(a^{q+2} / b_j\right) P, \ldots, \left(a^{2q} / b_j\right) P$$

$$\forall_{1 \leq j,k \leq q, k \neq j} \left(a \cdot s \cdot b_k / b_j\right) P, \ldots, \left(a^q \cdot s \cdot b_k / b_j\right) P$$

it must remain hard to distinguish $e(P, P)^{a^{q+1}} \in G_2$ from a random element in $R \in G_2$.

An algorithm $\mathcal{A}$ that outputs $z \in \{0, 1\}$ has advantage ϵ in solving decisional *q-parallel* BDHE in G_1 if

$$\left| \Pr\left[\mathcal{A}\left(\vec{y}, T = e(g, g)^{a^{q+1} s}\right) = 0\right] \right.$$

$$\left. - \Pr\left[\mathcal{A}\left(\vec{y}, T = R\right) = 0\right] \right| \geq \epsilon$$

We say that the (decision) *q-parallel* BDHE assumption holds if no polytime algorithm has a non-negligible advantage in solving the decisional *q-parallel* BDHE problem [2].

System Model and Definition

Model

There are four entities in the proposed scheme as follows.

User: The user downloads the shared data from Cloud Server.

Data owner: The data owner encrypts the shared data then uploads to Cloud Server.

Authority: The authority manages attributes in the system and publishes the parameters used for encryption. It generates a secret key that user's attributes are embedded and PRE keys used for re-encryption and updating secret key. The authority is trusted party.

Cloud Server: Cloud Server stores shared data. It re-encrypts encrypted shared data and update secret key by using PRE keys received from the authority. Similar to previous schemes [3,4], we assume Cloud Server to be curious-but-honest. That is, it will honestly execute the tasks assigned by legitimate parties in the system. However, it would like to learn information of encrypted shared data as much as possible.

Algorithm Definition

Our proposed scheme is composed of 8 algorithms: Auth.Setup, DO.Enc, Auth.Ext, U.Dec, Auth.ReKeyGen, C.ReEnc, C.ReKey, C.AddAtt.

Auth.Setup: The setup algorithm takes as input the security parameter and attribute universe description. It outputs the public parameters PK, master secret key MSK and the keys for granting an attribute J.

DO.Enc: The Encryption algorithm takes as input the public parameters PK, an LSSS access structure $\mathbb{A}$, and a message $\mathcal{M}$. It outputs a ciphertext CT.

Auth.Ext: The key extraction algorithm takes as input the master key MK, and a set of attributes S. It outputs a secret key SK and t_{ID}.

U.Dec: The decryption algorithm takes as input a secret key SK for a set S and a ciphertext CT for an access structure $\mathbb{A}$. If the set of attributes S satisfies the access structure $\mathbb{A}$, it outputs a message $\mathcal{M}$.

Auth.ReKeyGen: The re-encryption key generation algorithm takes as input the master key MK and a set of attributes γ for update. It outputs the redefined master key MK', the redefined public parameters PK', and the PRE (Proxy Re-Encryption) keys rk.

C.ReEnc: The re-encryption algorithm takes as input an attribute y for update, the ciphertext component D_i and a PRE key list RKL_y. It outputs the re-encryption ciphertext component D_i'.

C.ReKey: The key regeneration algorithm takes as input an attribute w for update, the secret key component K_w and the PRE key list RKL_w. It outputs the updated secret key component K_w'.

C.GrantAtt: The attribute grant algorithm takes as input an attribute v, the key for granting an attribute J_v, t_{ID} and the PRE key list RKL_v. It outputs secret key component K_v and redefines the key for granting an attribute J_v'.

Security Definition

We prove that unauthorized users and Cloud Server cannot decrypt ciphertext CT that was encrypted by the proposed scheme. Since we assume Cloud Server is honest, we do not consider active attacks from Cloud Server by colluding with unauthorized or revoked users. We define two attack models and security models as follows.

Attack Model 1 In this model, we assume an attack by unauthorized users. Security in this model is defined with the following game.

-Init. The adversary A submits the challenge access structure $\mathbb{A}$ to the challenger C.
-Setup. The challenger C runs setup algorithm and gives the public parameters PK to the adversary A.
-Phase1. The adversary can issue following query.

- Ext query : The adversary A submits a set of attributes S where S does not satisfy the access structure $\mathbb{A}$ to the challenger. The challenger C gives secret key corresponding S.
- Add query : The adversary A submits a set of attributes S' where $S \cup S'$ does not satisfy the challenge access structure $\mathbb{A}$. The challenger C gives the secret key component K_x corresponding to S'.

-Challenge. The adversary A submits two equal length messages M_0, M_1. The challenger flips a random coin b, and encrypts M_b under $\mathbb{A}$. The challenger gives ciphertext CT to the adversary A.
-Phase2. Phase1 is repeated.
-Guess. The adversary A outputs his guess b' of b.

The advantage of an adversary A in this game is defined as $\Pr\left[b' = b\right] - \frac{1}{2}$.

Definition 2 A ciphertext-policy attribute-based encryption scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.

Attack Model 2 In this model, we assume an attack by Cloud Server. Security in this model is defined with the following game.

-Init. The adversary A submits the challenge access structure $\mathbb{A}$ and version number ver^* to the challenger C.
-Setup. The challenger C runs setup algorithm and gives the public parameters PK and PRE key and the keys for granting an attribute J to the adversary A.
-Phase1. The adversary can issue following query.

- K_x query : The adversary A submits a set of attribute S. The challenger C gives secret key component K_x corresponding to S to the adversary A.

-Challenge. The adversary A submits two equal length messages M_0, M_1. The challenger flips a random coin b, and encrypts M_b under $\mathbb{A}$. The challenger gives ciphertext CT to the adversary A.
-Phase2. Phase1 is repeated.
-Guess. The adversary A outputs his guess b' of b.

Definition 3 A ciphertext-policy attribute-based encryption scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.

Our Scheme

Overview

The proposed scheme is based on Waters's scheme of CP-ABE [2]. Water's scheme supports any LSSS access structure. We apply the idea of attribute revocation

shown in [3] to the proposed scheme. In the proposed scheme, the attribute key is included in the ciphertext and secret key to delegate attribute revocation processes to Cloud Server. The attribute key is master key components corresponding to each attribute in the system. When user's attributes are revoked, the authority re-defines the attribute keys, and generates PRE keys for updating the attribute keys. Cloud Server re-encrypts ciphertext and updates secret key by updating attribute key by using PRE key. Each attribute is associated with version number for updating attribute key.

Cloud Server keeps user list UL, re-encryption key list RKL and the key for granting an attribute to secret key J. UL records user's *ID*, user's attribute information, secret key components, t_{ID}. t_{ID} is a random number that randomize each secret key to prevent users' collusion attack. t_{ID} should "bind" components of one user's key together so that they cannot be combined with another user's key components[2]. RKL records update history of attribute (version number) and PRE keys.

When granting attributes to users, Cloud Server generates user's secret key components correspond to granting attribute from t_{ID} and *J*, and sends secret key component to the user. The user joins secret key component to own secret key. Thus, it is possible to grant attributes to users without generation of new secret key by the authority.

Algorithm

Auth.Setup(U) The setup algorithm takes as input the number of system attributes U. It first chooses a group G_1 of prime order p, a generator $P \in G_1$. It then chooses random group elements $Q_1, ..., Q_U \in G_1$ that are associated with the U attributes in the system. In addition, it chooses two random $\alpha, a \in Z_p$, and random $Att_1, ..., Att_U \in Z_p$ as the attribute key.

The public parameters are

$$PK := \quad < P, e(P,P)^{\alpha}, aP, Q_1, ..., Q_U, T_1 = Att_1 P, ..., T_U >$$

The master key is $MK := < \alpha, Att_1, ..., Att_U >$.

The keys for granting an attribute are $J := < \{x, J_x = 1/Att_x\}_{1 \le x \le U} >$.

DO.Enc ($PK, (M, \rho), \mathcal{M}$) The Encryption algorithm takes as input the public parameters *PK*, an LSSS access structure (M, ρ), and a message $\mathcal{M}$. The function ρ associates rows of M to attributes. Let M be an $l \times n$ matrix. It first chooses a random vector $\vec{v} = (s, y_2, ..., y_n) \in Z_p$. For $\boldsymbol{i} = 1$ to l, it computes $\lambda_i := \vec{v} \cdot M_i$. It then chooses random $r_1, ..., r_l \in Z_p$ and outputs the ciphertext

$$CT := \quad < C, C', (C_1, D_1), ..., (C_l, D_l) >=$$

$$< Ke(P,P)^{\alpha s}, sP, \left(\lambda_1 (aP) - r_1 Q_{\rho(1)}, r_1 T_{\rho(1)}\right), ..., \left(\lambda_l (aP) - r_l Q_{\rho(l)}, r_l T_{\rho(l)}\right) >$$

with (M, ρ).

Auth.Ext (MK, S) The key extraction algorithm takes as input the master key *MK*, and a set of attributes *S*. It first chooses a random $t_{ID} \in Z_p$. It then outputs t_{ID} and the secret key

$$SK := < K, L, \forall x \in S\ K_x > =$$
$$< \alpha P + t_{ID}(aP), t_{ID}P, \forall x \in S\ (t_{ID}/Att_x)Q_x > .$$

U.Dec (SK, CT) The decryption algorithm takes as input a secret key SK for a set S and a ciphertext CT for access structure (M, ρ). Suppose that S satisfies the access structure and let I be defined as $I = \{i : \rho(i) \in S\}$. Then, let $\{\omega_i \in Z_p\}_{i \in I}$ be as set of consistants such that if $\{\lambda_i\}$ are valid shares of the secret s according to M, then $\sum_{i \in I} \omega i \lambda_i = s$.

The decryption algorithm first computes

$$\frac{e(C', K)}{\prod_{i \in I}\left(e(C_i, L)e\left(D_i, K_{\rho(i)}\right)\right)^{\omega_i}} =$$
$$\frac{e(P,P)^{\alpha s} e(P,P)^{ast_{ID}}}{\prod_{i \in I}\left(e(P,P)^{ta\lambda_i\omega_i}\right)} = e(P,P)^{\alpha s}.$$

It can then decrypt the message $\mathcal{M} = C/e(P, P)^{\alpha s}$.

Auth.ReKeyGen(MK, γ) The re-encryption key generation algorithm takes as input the master key MK and a set of attributes γ for update. For each $x \in \gamma$, it chooses random $Att'_x \in Z_p$ as the new attribute key, and computes $T'_x := Att'_x P$, $rk_{x \to x'} := \frac{Att'_x}{Att_x}$. It then replaces each Att_x of the master key component with Att'_x, and each T_x of public parameter with T'_x. It outputs the redefined the master key MK', the redefined public parameters PK', and the PRE keys $rk := \{x, rk_x\}_{x \in \gamma}$.

C.ReEnc($y(=\rho(i)), D_i, RKL_y$**)** The re-encryption algorithm takes as input an attribute y $(=\rho(i))$ for update, the ciphertext component D_i and a PRE key list RKL_y. It first checks version of attribute y. If y has the latest version, it outputs $\perp$ and exit. Let $Att_{y^{(n)}}$ be defined as an attribute key of the latest version of attribute y. It computes $rk_{y \leftrightarrow y^{(n)}} := rk_{y \leftrightarrow y'} \cdot rk_{y' \leftrightarrow y''} \cdots rk_{y^{(n-1)} \leftrightarrow y^{(n)}} = Att_{y^{(n)}}/Att_y$. Then, it outputs the re-encrypted ciphertext component $D'_i := rk_{y \leftrightarrow y^{(n)}} \cdot D_i = \left(Att_{y^{(n)}}/Att_y\right) \cdot r_i Att_y P = r_i Att_{y^{(n)}} P$.

C.ReKey $(w, K_{w,ID}, RKL_w)$ The key regeneration algorithm takes as input an attribute w for update, the secret key component K_w and the PRE key list RKL_w. It first checks version of attribute w. If w has the latest version, it outputs $\perp$ and exit. Let $Att_{w^{(n)}}$ be defined as the attribute key for the latest version of attribute w. It computes $rk_{w \leftrightarrow w^{(n)}} := rk_{w \leftrightarrow w'} \cdot rk_{w' \leftrightarrow w''} \cdots rk_{w^{(n-1)} \leftrightarrow w^{(n)}} = Att_{w^{(n)}}/Att_w$. It then outputs the updated secret key component $K'_w := rk^{-1}_{w \leftrightarrow w^{(n)}} \cdot K_w = (Att_w/Att_{w^{(n)}}) \cdot (\boldsymbol{t_{ID}}/Att_w)Q_w = (\boldsymbol{t_{ID}}/Att_{w^{(n)}})$ Q_w.

C.GrantAtt (v, J_v, t_{ID}, RKL_v) The attribute grant algorithm takes as input an attribute v, the key for granting an attribute J_v, t_{ID} and the PRE key list RKL_v. It first checks version of attribute v. Let $Att_{v^{(n)}}$ be defined as the attribute key for the latest version of attribute v. It first computes $rk_{v \leftrightarrow v^{(n)}} := rk_{v \leftrightarrow v'} \cdot rk_{v' \leftrightarrow v''} \cdots rk_{v^{(n-1)} \leftrightarrow v^{(n)}} = Att_{v^{(n)}}/Att_v$. It then outputs secret key component for $K_v := t_{\boldsymbol{ID}} \cdot rk^{-1}_{v \leftrightarrow v^{(n)}} \cdot J_v = t_{ID} \cdot (Att_v/Att_{v^{(n)}}) \cdot (1/Att_v)Q_v = (t_{ID}/Att_{v^{(n)}})Q_v$ and redefines the key for granting an attribute $J'_{\boldsymbol{v}} := rk^{-1}_{v \leftrightarrow v^{(n)}} \cdot J_v = (1/Att_{v^{(n)}})Q_v$.

We show the flow of our scheme in Fig 1. In Fig 1, γ denotes a set of user u's attributes which are revoked and β denotes a set of attributes that granting to user u.

Security Proof

We prove that unauthorized users and Cloud Server cannot decrypt ciphertext CT that was encrypted by using the proposed scheme.

Security Proof in the Attack Model 1

Theorem 1 Suppose the decisional *q-parallel* BDHE assumption holds and a challenge matrix of size is $l^* \times n^*$ where $l^* \times n^* \leq q$, our scheme is IND-CPA secure in the attack model 1.

Proof Suppose we have adversary A with non-negligible advantage ϵ against our scheme in the attack model 1. Moreover, suppose it chooses a challenge matrix M^* where both dimensions are at most q. We show how to build a simulator, B, that plays the decisional *q-parallel* BDHE problem.

Init. The simulator takes in a *q-parallel* BDHE challenge $\vec{y}, T$. The adversary gives the simulator B the challenge access structure (M^*, ρ^*), where M^* has n^* columns.

Setup. The simulator B generates the public parameter PK as follows. The simulator B chooses random $\alpha' \in Z_p$ and implicitly sets $\alpha = \alpha' + a^{q+1}$ by letting $e(P,P)^{\alpha} = e(aP, a^q P)e(P,P)^{\alpha'}$. It outputs public parameter $Q_1, \ldots, Q_U$ as follows.

1. For each x for $1 \leq x \leq U$ begin by choosing a random value z_x
2. Let X denote the set of indices i, such that $\rho^*(i) = x$.
3. The simulator B computes
$$Q_x = z_x P + \sum_{i \in X} \left\{ \left(a^2 M^*_{i,1}/b_i\right)P + \left(a^2 M^*_{i,2}/b_i\right)P + \cdots + \left(a^{n^*} M^*_{i,n^*}/b_i\right)P \right\}$$

Note that if $X = \emptyset$ then we have $Q_x = g^{z_x}$. Also note that the parameters are distributed randomly due to g^{z_x}. The simulator B randomly chooses attribute keys $t_x \in Z_p$ for

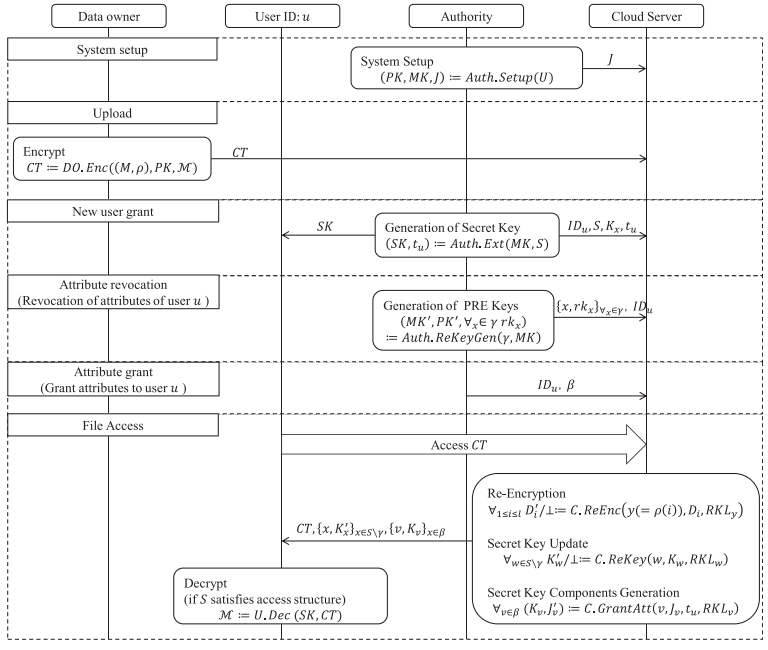

Figure 1 Flow of the proposed scheme.

$1 \le x \le U$ and computes public parameters $T_x = t_x P$. It gives the adversary A the public parameters $PK := (P, e(P,P)^{\alpha}, aP, Q_1, \ldots, Q_{Ld}, T_1, \ldots, T_U)$.

Phase1. The adversary A issues following queries:

Ext query : The adversary A submits a set of attributes S where S does not satisfy the access structure M^* to the challenger. The simulator first chooses a random $r \in Z_p$. Then it finds a vector $\vec{w} = (w_1, \ldots, w_{n^*}) \in Z_p^{n^*}$ such that $w_1 = -1$ and for all i where $\rho(i) \in S'$ we have that $\vec{w} \cdot M_i^* = 0$. The simulator B begins by implicitly defining t_{ID} as

$$r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$$

It performs this by setting $L = rP + \sum_{i=2,\ldots,n^*} w_i (a^{q+1-i} P) = t_{ID} P$. The simulator can compute K as

$$\alpha' P + arP + \sum_{i=2,\ldots,n^*} w_i \left(a^{q+2-i} P\right)$$

The simulator B computes $K_x \ \forall_x \in S$ as follows.

Case 1. If there is no i such that $\rho^*(i) = x$, it computes $K_x = (1/t_x) \cdot z_x L$.

Case 2. If there is i such that $\rho^*(i) = x$.

1) Let X be the set of all i such that $\rho^*(i) = x$.
2) It computes K_x as

$$z_x L + \sum_{i \in X} \sum_{j=1,\ldots,n^*} M_{i,j}^* \left[\left(a^j / b_i\right) rP + \sum_{\substack{k = 1,\ldots,n^* \\ k \neq j}} w_k \left\{\left(a^{q+1+j-k} / b^i\right) P\right\}\right]$$

3) It calculates $K_x = (1/t_x) K_x'$.

It gives the adversary A secret key $SK := (K, L, \forall x \in S \ K_x)$.

Add query : The adversary A submits a set of attributes S' where $S \cup S'$ does not satisfy the challenge access structure M^*.The simulator B computes $K_x \ \forall_x \in S'$ as follows.

Case1. If there is no i such that $\rho^*(i) = x$, it computes $K_x = (1/t_x) \cdot z_x L$.

Case2. If there is i such that $\rho^*(i) = x$.

1) Let X be the set of all i such that $\rho^*(i) = x$.
2) It computes K_x' as

$$z_x L + \sum_{i \in X} \sum_{j=1,\ldots,n^*} M_{i,j}^* \left[\left(a^j / b_i\right) rP + \sum_{\substack{k = 1,\ldots,n^* \\ k \neq j}} w_k \left\{\left(a^{q+1+j-k} / b^i\right) P\right\}\right]$$

3) It calculates $K_x = (1/t_x) K_x'$.

It gives the adversary A the secret key component $\{K_x\}_{\forall x \in S'}$.

Challenge. The adversary A submits two equal length messages $\mathcal{M}_0$, $\mathcal{M}_1$. The simulator B flips a random coin $b \in \{0, 1\}$. It computes $C = \mathcal{M}_b T \cdot e(sP, \alpha' P)$, $C = sP$. It choose random $y_2', \ldots, y_{n^*}' \in Z_p$ and the share the secret using the vector

$\vec{v} = \left(s, sa + y_2^{'}, sa^2 + y_3^{'}, \ldots, sa^{n-1} + y_{n^*}^{'}\right) \in Z_p^{n^*}$. In addition, it choose random values $r_1^{'}, \ldots, r_l^{'} \in Z_p$.

For $i = 1, \ldots, n^*$, we define R_i as the set of all $k \neq i$ such that $\rho^*(i) = \rho^*(k)$. The challenge ciphertext components are then generated as

$$D_i = -r_i^{'} T_{\rho^*(i)} - sb_i T_{\rho^*(i)}$$

$$C_i = h_{\rho^*(i)}^{r_i^{'}} + \left\{ \sum_{j=2,\ldots,n^*} M_{i,j}^* y_j^{'}(aP) \right\} - z_{\rho(i)}(b_i \cdot s)P + \{ \sum_{k \in R_i} \sum_{j=1,\ldots,n^*} M_{k,j}^* \left(a^j \cdot s \cdot (b_i / b_k)P\right) \}$$

It gives the adversary A the challenge ciphertext $CT^* = \left(C, C^{'}, (C_1, D_1), \ldots, (C_{l^*}, D_{l^*})\right)$.

Phase2. Phase 1 is repeated.

Guess. The adversary A will eventually output a guess b'. of b. The simulator then outputs 0 to guess that $T = e(P,P)^{a^{q+1}s}$ if $b'. = b$; otherwise, it outputs 1 to indicate that it believes T is a random group element $R \in G_2$.

When T is a tuple the simulator B gives a perfect simulation so we have that

$$\Pr = \left[\mathrm{B}\left(\vec{y}, T = e(P,P)^{a^{q+1}s}\right) = 0 \right] = \frac{1}{2} + \epsilon.$$

When T is a random group element the message $\mathcal{M}_b$ is completely hidden from the adversary and we have $\Pr = \left[\mathrm{B}\left(\vec{y}, T = R\right) = 0 \right] = \frac{1}{2}$. Therefore, the simulator B can play the decisional *q-parallel* BDHE game with non-negligible advantage.

Security Proof in the Attack Model 2

Theorem 2 Suppose Waters's scheme [2] is IND-CPA secure, our scheme is also IND-CPA secure in the attack model 2.

Proof Suppose we have adversary A with non-negligible advantage ϵ against our scheme in the attack model 2. Moreover, suppose it chooses a challenge matrix M^* where both dimensions are at most q. We prove there is an simulator B which has advantage at least ϵ against Waters's scheme simulator (Given input, it responds according to algorithms of the Waters's scheme).

Init. The adversary A submits the challenge access structure (M^*, ρ^*) where M^* has n^* columns and version number ver^* to the simulator B. The simulator B submits the challenge access structure (M^*, ρ^*) to the Waters's scheme simulator.

Setup. The simulator B receives public parameters $PK^{'} := (P, e(P,P)^{\alpha}, aP, Q_1, \ldots, Q_U)$ from the Waters's scheme simulator. It randomly chooses attribute keys $t_x \in Z_p$ for $1 \leq x \leq U$ and computes public parameters $T_x = t_x P$. It computes the key for granting an attribute $J := \{(1/t_1)Q_1, \ldots, (1/t_U)Q_U\}$. Then, the simulator B computes PRE keys and public parameters T_x for each version as follows. For x $(1 \leq x \leq U)$, for $1 \leq k \leq ver^* - 1$, the simulator B randomly chooses PRE keys $rk_{x^{(k)} \to x^{(k+1)}} \in Z_p$ and computes public parameters $T_{x^{(k+1)}} = rk_{x^{(k)} \leftrightarrow x^{(k+1)}} T_{x^{(k)}}$. $(k+1)$ and (k) denote the version number of PRE keys and public parameter. The simulator B gives the adversary A the public parameter $PK := (P, e(P,P)^{\alpha}, aP, Q_1, \ldots, Q_U, T_1, \ldots, T_U)$, the key for granting an attribute J and all PRE keys.

Phase1. The adversary A issues following queries:

K_x query : The adversary A submits a set of attributes S for version k, $1 \le k \le ver^* - 1$. we denote $V_{x^{(k)}} = \prod_{i=2}^{k} rk_{x^{(i-1)} \leftrightarrow x^{(i)}}$. The simulator B randomly chooses $t_{ID} \in Z_p$ and computes $\forall x \in S\ K_x = (t_{ID}/t_x \cdot V_{x^{(k)}})Q_x$. It gives the adversary A the secret key components $\{K_x\}_{\forall x \in S}$.

Challenge. The adversary A submits two equal length messages $\mathcal{M}_0$, $\mathcal{M}_1$, then the simulator B submits them to the Waters's simulator. The Waters's simulator flips a random coin $b \in \{0, 1\}$ and computes ciphertest $CT' := (C, C', (C_1, D_1), ..., (C_l, D_l)) \leftarrow Enc(PK, (M^*, \rho^*), \mathcal{M}_b)$. The simulator B receives the ciphertext CT', then it computes $CT := \left(C, C', \left(C_1, V_{\rho(1)^{(ver^*)}} D_1\right), ..., \left(C_l, V_{\rho(l)^{(ver^*)}} D_l\right)\right)$ from the ciphertext CT'. It gives the adversary A the cipertext CT.

Phase2. Phase 1 is repeated.

Guess. The adversary A outputs will eventually output a guess b' of b. The simulator B outputs b' as its guess.

The simulation above shows there is a simulator B that has advantage at least ϵ ageinst Waters's scheme simulator if there is an adversary A that has advantage ϵ against our scheme.

Result and discussion

In Table 1, we give two comparisons of the proposed scheme with the schemes of [3,4] that can revoke the only specified attributes. The first comparison is in terms of the size of the public key (PK), the secret key (SK), the ciphertext (CT), and the re-encryption key (RK). The second comparison is in terms of the computation amount of encryption (Enc), secret key generation (Ext), re-encryption (Re-enc), decryption (Dec), and secret key update (Re-key). As to the size of the public key, the scheme of [4] has the smallest one, followed by the proposed scheme. As for the size of the secret key,

Table 1 Key Size, Ciphertext Size and Computation Amount

	Yu etal's scheme [3]	Hur et al's scheme [4]	The proposed scheme
PK	$(3\|U\|+1) \times \|G\| + \|G_T\|$	$2 \times \|G\| + \|G_T\|$	$(2\|U\|+2) \times \|G\| + \|G_T\|$
SK	$(2\|U\|+1) \times \|G\|$	$(2\|S\|+1) \times \|G\| + \log\|N\| \times \|K\|$	$(\|S\|+2) \times \|G\|$
CT	$(\|U\|+1) + \|G\| + \|G_T\|$	$(2\|I\|+1) \times \|G\| + \|G_T\|$	$(2\|I\|+1) \times \|G\| + \|G_T\|$
RK	$r\|U\| \times \|Z_p\|$	$(2\|N\|-1) \times \|K\|$	$r\|U\| \times \|X_p\|$
Enc	$(\|U\|+2) \times exp$	$(2\|I\|+2) \times exp$	$(2\|I\|+2) \times exp$
Ext	$(2\|U\|+1) \times exp$	$(2\|S\|+2) \times exp$	$(\|S\|+2) \times exp$
Re-enc	$\|R_{CT}\| \times exp$	$\|R_{CT}\| \times exp$	$\|R_{CT}\| \times exp$
Re-key	$\|R_{SK}\| \times exp$	$\|R_{SK}\| \times exp$	$\|R_{SK}\| \times exp$
Dec	$(\|U\|+1) \times \hat{e}$ $+(\|U\|1) \times exp$	$(2\|R\|+1) \times \hat{e}$ $+(2\|R\|+2) \times exp$	$(2\|R\|+1) \times \hat{e}$ $+(2\|R\|+2) \times exp$

Exp:ex ponentiation in G, $\hat{e}$: bilinear pairing,
$|U|$: the number of attributes defined in the system,
$|S|$: the number of attributes in user's key,
$|R|$ the number of user's attributes satisfying an acces structure,
r: the number of times the attribute revocation event occurs,
$|R_{SK}|$: the number of updated attributes (secret key),
$|R_{CT}|$: the numbet of updated attributes (ciphertext), $|N|$: the number of total user,
$|I|$: the number of attributes am acces structure, $|K|$: size of the common key.

Table 2 Comparison of Schemes

	Yu etal's scheme [3]	Hur et al's scheme [4]	Liang et al's scheme [5]	The proposed scheme
Suporting acces policy type	'AND'	'AND', 'OR'	Any LSSS	Any LSSS
Attibute level user revoction	Possible	Possible	Impossible	Possible
Grant attributes to users	The authority generates a new secret key	The authority generates a new secret key	The authority generates a new secret key	Cloud server adds attributes to user's secret key

the proposed scheme has the smallest one. Both the proposed scheme and the scheme of [4] have equally the smallest size ciphertexts. As to the size of the re-encryption key, if there are users more than the number of attributes, both the proposed scheme and the scheme of [3] have the equally smallest one.

As for the computation amount of encryption and decryption, the proposed scheme and the scheme of [4] have the equally smallest. As to the computation amount of secret key generation, the proposed scheme has the smallest. Finally, as to the computation amount of re-encryption and secret key update, all schemes have the same.

The differences, in terms of the requirements in Section I, between the proposed scheme and the schemes of [3-5] are summarized as shown in Table 2.

Conclusion

This paper proposed a ciphertext-policy attribute-based encryption scheme delegating attribute revocation processes to Cloud Server by proxy re-encryption. Cloud Server re-encrypts a ciphertext and updates a secret key by updating attribute key with PRE key for updating the attribute keys.

The proposed scheme meets three requirements as follows; First, the proposed scheme supports any LSSS access structure. Second, the authority can only revoke specified attribute by updating attribute key included in ciphertext corresponding to his attributes which are revoked. Finally, when granting attributes to a user, generation of a new secret key becomes unnecessary because Cloud Server generates secret key components corresponding to granting attributes.

The proposed scheme is secure against attack by unauthorized users and Cloud Server. Our future direction is to implement the proposed scheme and confirm its feasibility.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
TN designed the study and drafted the manuscript. MM and YS conceived of the study, participated in the design and drafting the article and revising it critically for intellectual content. They reviewed and approved the final, submitted version. All authors read and approved the manuscript.

Authors' Information
TN received B.E. degree from Nagoya Institute of Technology, Japan, in 2013. He is a graduate student of the institute. His current research interests include information security, cryptography. He received DICOMO2013 symposium Paper Awards and Presentation Awards in 2013. He is a member of IPSJ.
MM received B.E. and M.E. degrees from Ehime University, Japan, in 1993 and 1995 respectively. She received Ph.D degree in Engineering from the University of Tokushima, Japan in 2002. From 1995 to 1998 she was an assistant professor at the Department of Management and Information Science, Kagawa junior college, Japan. From 1998 to 2002 she was a research associate of the Department of Information Science and Intelligent Systems, the University of Tokushima, Japan. From 2003 to 2008 she was a lecturer of the same department. Since 2008, she has been an

associate professor at the Information and Multimedia Center, Gifu University, Japan. Her research interests are in coding theory, information security and cryptography. She is a member of IEEE and a senior member of IEICE.
YS received B.E. and M.E. degrees from Ehime University, Japan, and Ph.D degree from the University of Tokushima, Japan, in 1995, 1997, and 2000, respectively. From 2002 to 2006 he was a lecturer at the Department of Informatics, Kinki University, Japan. From 2006 to 2013 he was an associate professor at the Department of Computer Science and Engineering, Nagoya Institute of Technology, Japan. Since 2013, he has been an associate professor at the Department of Electrical and Electronic Engineering, Kobe University, Japan. His current research interests include information security, cryptography, computer network, and knowledge sharing and creation support. He received the SCIS 20th Anniversary Award and the SCIS Paper Award from ISEC group of IEICE in 2003 and 2006, respectively. He is a member of IEEE, ACM and a senior member of IEICE, IPSJ.

Author details
[1]Nagoya Institute of Technology, Nagoya, Japan. [2]Gifu University, Gifu, Japan. [3]Kobe University, Kobe, Japan.

Received: 30 October 2013 Accepted: 6 March 2015
Published online: 22 March 2015

References

1. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: Paper presented at the 2007 IEEE Symposium on Security and Privacy, Oakland., 20–23 May 2007
2. Waters B (2011) Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Paper presented at the 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina., 6–9 March 2011
3. Yu S, Wang C, Ren K, Lou W (2010) Attribute based data sharing with attribute revocation. In: Paper presented at the 5th ACM Symposium on Information. Computer and Communications Security, Beijing, 13 April 2010
4. Hur J, Nor D.K (2011) Attribute-based access control with efficient revocation in data outsourcing systems. doi:10.1109/TPDS.2010.203.
5. Liang X, Lu R, Lin X, Shen X (2011) Ciphertext policy attribute based encryption with efficient revocation. http://bbcr.uwaterloo.ca/~x27liang/papers/abe%20with%20revocation.pdf.
6. Beimei A (1996) Secure schemes for secret sharing and key distribution. Dissertation, Israel Institute of Technology.