Journal of Information Security and Applications 48 (2019) 102355

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Attributes revocation through ciphertext puncturation

Hongyong Jia[a,*], Yue Chen[b], Yan Li[a], Xincheng Yan[b], Fenlin Liu[b], Xiangyang Luo[b], Bo Wang[c]

[a] Zhengzhou University, No. 97, Wenhua Road, Zhengzhou City, Henan Province, 450002, P.R. China
[b] Zhengzhou Science and Technology Institute, No. 62, Science Avenue, National High & New Technology Industries Development Zone, Zhengzhou City, Henan Province, 45000, P.R. China
[c] State University of New York at Buffalo, Flint Entrance, Amherst, NY 14260, USA

ARTICLE INFO

Article history:
Available online xxx

Keywords:
Attribute based encryption
Revocation
Puncturation
Non-monotonic
Linear secret sharing

ABSTRACT

In order to solve the difficult issue of attribute revocation in the attribute based encryption scheme, a novel method of revoking attributes through ciphertext puncturation is proposed. In this method, a ciphertext puncturation algorithm is designed and the "NOT" operator's ability to negate attributes in the non-monotonic access policy is utilized to revoke attributes. First, a non-monotonic access policy is constructed from the attributes revocation list. Then the ciphertext is punctured with this policy using the re-randomization technique. Finally, double policies exist in the ciphertext to implement access control. Without any interaction for private key update, the private key including any revoked attributes directly loses its decryption ability due to the puncturation of the ciphertext with the non-monotonic access policy containing revoked attributes. After puncturation, the ciphertext gets forward secrecy and attribute revocation is achieved. Theoretic analysis indicates that the proposed scheme maintains the security level of the attribute based encryption scheme with non-monotonic access policy and effectively completes attributes revocation.

1. Introduction

Attribute-based Encryption (ABE) is regarded as one of the most suitable technologies for secure data access control in cloud storage systems [1,2]. It allows data owners to define access policies and encrypt the data under these policies. Only users whose attributes satisfying these access policies can decrypt the data ciphertext. The mechanism of attribute based encryption greatly enriches the flexibility of encryption policies and the descriptiveness of user access privileges [3–5]. The encryption model is extended from one-to-one pattern into one-to-many pattern. In an access control system implemented using attribute based encryption, attributes revocation should be considered for the ever changing user privilege. Attribute revocation is a challenging issue in attribute based encryption schemes since a ciphertext maybe decrypted by several users and each attribute maybe shared by multiple users [6]. A large number of ABE schemes with attribute revocation had been proposed. These schemes can be divided into three kinds according to the attribute revocation mechanism: schemes using proxy re-encryption, schemes where attribute revocation information is embedded into ciphertext during encryption and schemes where decryption process is divided between the cloud and the user.

In the first kind of scheme, the cloud re-encrypted the ciphertext utilizing keys given by the data owner and changed access policies in the ciphertext to revoke attributes. Ref. [6] proposed a revocable attribute based encryption scheme using proxy re-encryption. It is needed to update private keys not affected by revoked attributes. Ref. [7] proposed a scheme to update user's attributes by the cloud. The cloud re-encrypted the ciphertext and generated new private keys to users whose attributes had been updated by the cloud. Ref. [8] updated the ciphertext through proxy re-encryption and then revoked users cannot decrypt the ciphertext in the cloud. Ref. [9] and [10] also proposed similar schemes using proxy re-encryption to revoke attributes. In recently proposed schemes using proxy re-encryption, decryption process is divided into two parts. Only with the help of the cloud, can the user decrypt the ciphertext. Ref. [11] proposed a revocable scheme. In this scheme, the data owner sent the ciphertext to the cloud and the cloud re-encrypted the ciphertext according to authenticated valid users. Then the cloud deleted revoked users from the valid user list. The cloud sent partially decrypted ciphertext to the user and only non-revoked users can decrypt the ciphertext. Ref. [11] only supported user revocation not attribute revocation.

* Corresponding author.
E-mail address: jiahy_pla@126.com (H. Jia).

https://doi.org/10.1016/j.jisa.2019.06.011
2214-2126/

Ref. [12] let the cloud use proxy re-encryption key being specially manipulated to reduce trust on the cloud. Schemes based on proxy re-encryption has good flexibility in changing ciphertext policy. With the help of the cloud, workload for the user side to finish attribute revocation can be greatly reduced. But attribute revocation schemes using such method involve using transformation key to conduct proxy re-encryption. This will increase a large mount of key management workload for the cloud. It is also needed to interact with users to update private keys.

In the second kind of revocation scheme, attribute revocation list information was used during encryption to finish attribute revocation. The encryptor were required to get revocation list information and embed them into the ciphertext. Then when a user appeared in the revocation list, he couldn't decrypt the ciphertext even if his attribute set satisfied the policy in the ciphertext. There is no need to update user's private key. Ref. [13] proposed such a scheme by using broadcast encryption to distribute revocation list information. Ref. [14] proposed a similar key-policy attribute based encryption. The difference is that [14] only needed the encryptor to know identities of revoked users. Ref. [15] proposed a revocable ciphertext-policy attribute based encryption where the encryptor only needed to know the identities of the revoked users but the security model is relatively weak. Ref. [16] proposed another revocable CP-ABE scheme using similar approach. Ref. [17] proposed a KP-ABE construction with similar approach using multilinear maps. Ref. [18] proposed a CP-ABE construction. Again they employed similar approach but using matrix representation for users. Ref. [19] proposed another ABE scheme that supported revocability with this approach. This time, they used subset difference technique to achieve the purpose. Ref. [20] proposed a revocable ABE scheme using this approach. They used expire date of private key to reduce the length of revoked user list and then reduced the ciphertext length. Ref. [21] introduced user revocation centre (URC) in the revocable ABE scheme, and outsourced the revocation tasks to URC. Users need not to master the latest revocation list for encrypting, and need not to pay any additional computing for revocation. URC could update the ciphertext for users. ABE schemes with attribute revocation based on this approach have no need to update private keys of revoked users. But revoked users can decrypt the ciphertext generated before revocation. So schemes using this approach do not achieve ciphertext forward secrecy. In addition, these schemes only support user level revocation and don't support attribute revocation.

In the third kind of scheme, user's decryption capability was divided into two parts controlled by the cloud and the user separately. Ref. [22] and [23] proposed revocable ABE schemes based on decryption splitting independently. In their schemes, the decryption was split into two parts. A complete decryption requires both parts. User revocation is achieved by instructing the cloud server not to offer the needed assistance to the user. Ref. [24] used similar approach. They further reduced the trust on cloud but the risk of collusion between the cloud and users increased. In such schemes, there was no need to update user's private key, but the ciphertext didn't get updated, so these schemes lack ciphertext forward secrecy and they only achieve user level revocation. This kind of scheme didn't use proxy re-encryption while the first kind of scheme which also employed decryption division used proxy re-encryption techniques to achieve revocation.

From the analysis above, it can be found that existing revocable ABE schemes can flexibly change ciphertext access policy and achieve different level of revocation. With the assistance of the cloud, a large mount of revocation workload for users can be reduced. But in some special applications, where ciphertext forward secrecy and zero interaction for private key update are required when revoking attributes, there is no proper revocable ABE schemes available. In order to solve this issue, a novel scheme is proposed based on a ciphertext-policy ABE supporting non-monotonic access policy. In this scheme, attribute revocation list is transformed into a non-monotonic access policy. Then the non-monotonic policy is inserted into the ciphertext to merge with existing policy using public information and re-randomization technique. Finally, there are double access policies in the ciphertext to implement access control. Attributes revocation is achieved by utilizing the ability of negating attributes of the "NOT" operator in the non-monotonic access policy in the ciphertext.

2. Related works

The proposed scheme is based on a non-monotonic ciphertext-policy ABE scheme in [25]. Ref. [26] proposed a non-monotonic key-policy ABE scheme for the first time. The key idea in [26] was a method which could transform a non-monotonic access policy into a monotonic one and this method is called OST method. The basic concepts in the OST method is defined as follows:

For any non-monotonic access policy P, the attributes which may appear in the policy form a set called S, each attribute in S is ordinary and called positive attribute. We can find a corresponding monotonic access policy P' where the appeared attributes form a set called $M(S)$. Any attribute in $M(S)$ maybe negative as x' corresponding to attribute described by the operator "NOT" or positive denoted as x corresponding to attribute described by other operators. All the attributes that may appear in all kinds of policies forms a set called Z.

The transformation process in the OST method is defined as follows:

(1) given a non-monotonic access policy P, finding an attribute set S which can satisfy policy P and every attribute in set S is positive.
(2) computing the extended attribute set $M(S)$. For every attribute in the set S, it is added into the extended set $M(S)$.
(3) For every attribute in the whole attribute set Z, if it is not in set S, then it is changed into a negative attribute and added into the set $M(S)$.
(4) after completing the construction of set $M(S)$, a monotonic access policy P' can be found which is satisfied by the set $M(S)$. So we can transform the non-monotonic access policy P into a monotonic one P'.

The set $M(S)$ consists of attributes from the set S and negative attributes transformed from attributes which are not in S but in Z. So the set $M(S)$ is computed as: $M(S) = S \bigcup \{x' \mid x \in (Z/S)\}$.

Although [26] proposed the first non-monotonic ABE scheme, it only supported key-policy ABE scheme and is not efficient. Ref. [25] proposed an efficient ciphertext-policy ABE scheme supporting non-monotonic access structure which was called Yamada scheme. Compared with the scheme in [26], the Yamada scheme improved efficiency and policy expressiveness and supported fully unbounded size of attribute sets and access policies. The master public key of this scheme consisted of only constant number of group elements. The Yamada scheme consisted of four algorithms as follows:

Setup(λ). The setup algorithm chooses bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$, and picks $g \xleftarrow{s} \mathbb{G}$, $b, \alpha \xleftarrow{s} \mathbb{Z}_p$ and chooses $H, U, V, W \xleftarrow{s} \mathbb{G}$, then it sets $V' = U^b$. Finally, the setup algorithm outputs the master public key $\mathbf{mpk} = (g, H, U, V, V', W, e(g, g)^\alpha)$ and the master secret key $\mathbf{msk} = (\alpha, b)$.

KeyGen(msk,mpk,ω). The *KeyGen* algorithm generates a private key for a user with the attribute set $\omega = \{\omega_1, \omega_2, \ldots, \omega_k\} \subset \mathbb{Z}_p$. The algorithm first chooses $r, r_1, r_2, \ldots, r_k \xleftarrow{s} \mathbb{Z}_p$ and random values

$r'_1, \ldots, r'_k \in \mathbb{Z}_p$ such that $r'_1 + r'_2 + \ldots + r'_k = r$. Then it computes the private key as:

$$SK_\omega = \left(D_1 = g^\alpha W^r, D_2 = g^r, \left\{ \begin{matrix} K_{i,1} = V^{-r}(U^{\omega_i}H)^{r_i}, K_{i,2} = g^{r_i} \\ K'_{i,1} = (U^{b\omega_i}H^b)^{r'_i}, K'_{i,2} = g^{br'_i} \end{matrix} \right\} i \in [k] \right)$$

Encrypt(mpk,M,$\tilde{\mathbb{A}}$). The *Encrypt* algorithm needs the master public key **mpk**, a message $M \in \mathbb{G}_T$ and a non-monotonic access structure $\tilde{\mathbb{A}}$. The algorithm then converts the non-monotonic access structure $\tilde{\mathbb{A}}$ to a monotonic one $\mathbb{A}$ over an attribute set $\mathcal{P}$ which composed of negated attributes and non negated attributes according to the method in [26]. Next, the algorithm constructs a linear secret sharing scheme (L, π) where L is a $\ell \times m$ access matrix and π is a share assignment function which assigns share to corresponding attribute. Then it picks a random vector $S = (s, s_2, \ldots, s_m) \xleftarrow{s} \mathbb{Z}_p^m$ and computes share of s for $\pi(i)$ by $\lambda_i = \langle L_i \cdot s \rangle$ for $i = 1, \ldots, \ell$. Finally, the algorithm chooses a random value $t_i \xleftarrow{s} \mathbb{Z}_p$ and compute the ciphertext as the follows:

$$C_0 = M \cdot e(g,g)^{\alpha \cdot s},$$

$$C_1 = g^s \cdot \left\{ \begin{matrix} C_{i,1} = W^{\lambda_i} V^{t_i}, C_{i,2} = (U^{x_i}H)^{-t_i}, C_{i,3} = g^{t_i} & \pi(i) = x_i \\ C_{i,1} = W^{\lambda_i} V'^{t_i}, C_{i,2} = (U^{x_i}H)^{-t_i}, C_{i,3} = g^{t_i} & \pi(i) = x'_i \end{matrix} \right\}$$

The final output is $C = (C_0, C_1, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i \in [\ell]})$.

Decrypt(mpk, CT, sk_ω). The *Decrypt* algorithm is run by a user with private key sk_ω and ω is the attribute set belongs to this user. The monotonic policy $\mathbb{A}$ embedded in the ciphertext CT is transformed from the non-monotonic policy $\tilde{\mathbb{A}}$ used in the encryption algorithm. Assume $\tilde{\mathbb{A}}$ is satisfied by the attribute set ω, so the user can decrypt the ciphertext. It is needed to transform the attribute set ω into $\omega' = N(\omega)$ according the OST method in [26]. It can be decided that the monotonic access policy $\mathbb{A}$ is satisfied by the attribute set ω'. Let $I = \{i | \pi(i) \in \omega'\}$, the user can efficiently compute reconstruction coefficients $\{(i, \mu_i)\}_{i \in I} = Recon_{L,\pi}(\omega')$ such that $\sum_{i \in I} \mu_i \lambda_i = s$. Next, the user parses the ciphertext CT as $(C_0, C_1, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i \in \ell})$ and the user's private key is

$$sk_\omega = (D_1, D_2, \{K_{i,1}, K_{i,2}, K'_{i,1}, K'_{i,2}\}_{i \in [k]}).$$

Then the user can compute $e(g,g)^{r \cdot \lambda_i}$ for each $i \in I$ as follows:

$$\left\{ \begin{matrix} e(C_{i,1}, D_2) \cdot e(C_{i,2}, K_{\tau,2}) \cdot e(C_{i,3}, K_{\tau,1}) \to e(g,W)^{r \cdot \lambda_i} & \pi(i) = x_i \\ e(C_{i,1}, D_2) \cdot \prod_{j \in [k]} (e(C_{i,3}, K'_{j,1}) \cdot e(C_{i,2}, K'_{j,2}))^{\frac{1}{x_i - w_j}} \to e(g,W)^{r \cdot \lambda_i} & \pi(i) = x'_i \end{matrix} \right\}$$

where τ is the index of such that $\omega_\tau = x_i$. Such τ exists if $i \in I$ and $\pi(i)$ is a non-primed attribute. Next, the user computes

$$\begin{aligned} & e(C_1, D_1) \cdot \prod_{i \in I} (e(g,W)^{r\lambda_i})^{-\mu_i} \\ &= e(g^s, g^\alpha) e(g,W)^{sr} e(g,W)^{-r\sum_{i \in I} \mu_i \lambda_i} \\ &= e(g,g)^{\alpha \cdot s} \end{aligned}$$

Finally, the user can recover the message $M = C_0 / e(g,g)^{\alpha s}$

The Yamada scheme can revoke attribute's access privilege through non-monotonic access policy during encryption. The encryptor can design a non-monotonic access policy which use "NOT" operator to describe revoked attributes. Users having such revoked attributes cannot decrypt the ciphertext. But this application require attribute revocation information during encryption. In a large number of secure data sharing scenarios, attribute revocation information is released after ciphertext generation. The ciphertext policy is fixed and cannot be changed after generation. So the simple application of Yamada scheme cannot achieve dynamic attributes revocation.

An attributes revocation scheme including a ciphertext puncturation algorithm is designed to solve this issue. In this scheme, a third party such as the cloud constructs a non-monotonic access policy from the attribute revocation list and insert it into the ciphertext using public information and re-randomization technique. Finally, there are double access policies implementing access control in the ciphertext. When a user's private key consisting of any revoked attribute, the user will lose decryption capability for the ciphertext directly.

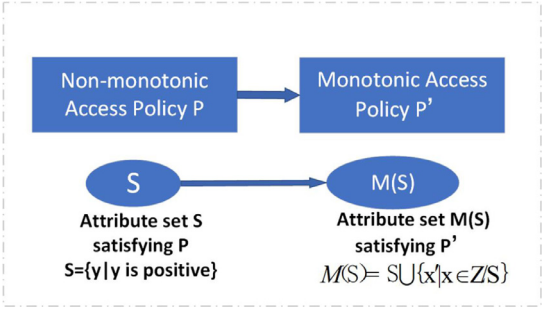

Fig. 1. The OST method.

There are two schemes in [27,28] which were similar to this attributes revocation scheme. In [27], the revocable storage was first proposed. A third party can modify the ciphertext and change the access policy into a more restrictive one to revoke access privileges of some users using public information. But this scheme needed updating private keys of non-revoked users and couldn't be used directly to revoke attributes. In [28], the third party can revoke some users' access rights from current ciphertext using public information. But this scheme was based on identity based broadcast encryption and could only support identity revocation not attributes revocation.

3. System model and proposed scheme

In the first part of this section, the system model of the proposed scheme is given. The data owner encrypts the data with ABE scheme [25] and send the ciphertext to the cloud for secure data sharing with other valid users. The cloud puncture the ciphertext to revoke attribute's privilege when receiving attribute revocation request. In the second part, the concrete ciphertext puncturation scheme is given.

3.1. System model

There are four roles in the system. The data owner (DO) owns data for share, encrypts data and sends the encrypted data to the cloud through public channel. The cloud authority center (CAC) is in charge of the security transaction of the cloud. It's main task includes initializing cryptographic system in the cloud, issuing the main public key information, securely saving the main private key, generating and sending private keys for registered users through secure channel, issuing attribute revocation list (ARL). The cloud data management center (CDMC) is responsible for data storage management. It can puncture ciphertext according to the attribute revocation list issued by the CAC. The data user (DU) owns its attribute set and private key generated by the CAC. The DU can access the ciphertext stored by the CDMC. The system model is given in Fig. 1.

3.2. Attributes revocation scheme

1 System Initialization. The CAC in the cloud chooses a security parameter λ, calls the Yamada's initialization algorithm $Setup(\lambda) \to (MPK, MSK)$ to generate the main public key MPK and private key MPK. The CAC publishes the main public key and securely store the main private key (Fig. 2).

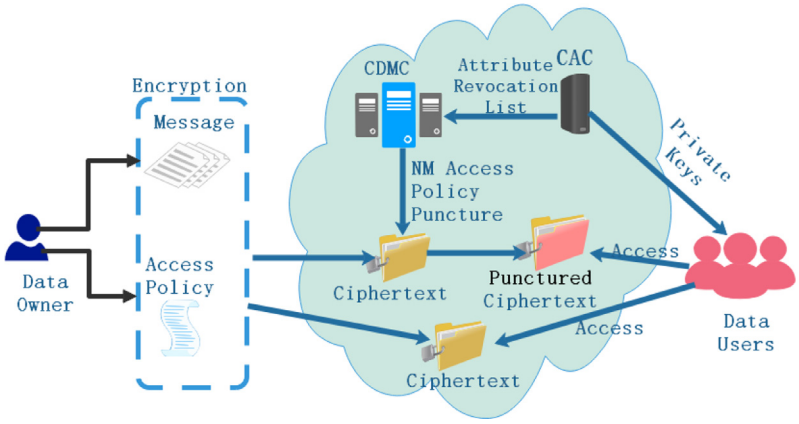

Fig. 2. System model.

2 Private Key Distribution. The CAC authenticates the identity and attribute set of the registered user. Then the CAC calls Yamada's private key generation algorithm $KeyGen(MPK, MSK, \omega) \rightarrow (SK_\omega)$ to generate private key SK_ω for the user and send the private key to the user through secure channel.

3 Data Sharing. The data owner sets access policy P according to the need for sharing plaintext M. Then the data owner calls the Yamada's encryption algorithm $Encrypt(mpk, M, \tilde{\mathbb{P}}) \rightarrow CT$ to generate ciphertext CT and sends the ciphertext to the cloud for data sharing.

4 Data Access. When the registered data user needs to access the ciphertext CT stored in the cloud, he calls the Yamada's decryption algorithm to get plaintext data M. $Decrypt(MPK, CT, sk_\omega) \rightarrow M$

5 Ciphertext Puncturation. The CAC decides which attribute should be revoked according to security requirement and constructs attribute revocation list. The CAC sends ARL to the CDMC. The CDMC calls ciphertext puncturation algorithm $CTPuncture(CT, MPK, ARL) \rightarrow CT'$ to generate new ciphertext CT' by puncturing the ciphertext with non-monotonic access policy from ARL. The CDMC then deletes the old ciphertext CT. Once the private key of a user contains any attribute in the ARL, this user will lose decryption capability immediately. Attributes revocation is achieved without updating user's private key. The puncturation algorithm $CTPuncture(CT, MPK, ARL) \rightarrow CT'$ is as follows:

(1) Constructing a non-monotonic access policy from the parameter ARL. Supposing the attribute revocation list is $ARL = \{A, B, C\}$, the whole attribute set in the system is $\{A, B, C, D, E, F, G\}$. Then the non-monotonic access policy is constructed as $P = \neg A \wedge \neg B \wedge \neg C$. The attribute set S that satisfies the policy P can be set as $S = \{D, E, F, G\}$ã.
(2) Changing the non-monotonic access policy P into a monotonic access policy P'. According to the OST method, the extended attribute set can be computed as $M(S) = \{D, E, F, G, A', B', C'\}$ which satisfies the monotonic access policy P'. So the monotonic policy can be set as $P' = A' \wedge B' \wedge C'$. In this policy, A', B', C' are negative attributes in the monotonic access policy representing attributes described by "NOT" operator in the non-monotonic access policy.
(3) Merging with existing policy in the ciphertext. The algorithm first finds the access policy P_{CT} accompanying with the ciphertext and connects the policy P' with P_{CT} using logic operator $\wedge$ to form a new access policy P_{new}. Next, computes the access matrix M_{CT} corresponding to the policy P_{CT} and update the matrix M_{CT} to get the matrix M_{new} corresponding to the policy P_{new} according to the inserted policy P'. Detailed process of matrix update can be found in [25]. If there is a negative attribute X' in the access policy P' and there is a positive attribute X in the access policy P_{CT}, then the positive attribute X in the newly formed policy P_{new} will be deleted.
(4) M' is a $\ell' \cdot m$ matrix corresponding to a linear secret sharing scheme (L', π'). First, the algorithm chooses a random vector $S' = (s', s'_2, \ldots, s'_m) \xleftarrow{s} \mathbb{Z}_p^m$ and computes share of s' for $\pi'(i)$ by $\lambda'_i = < L'_i \cdot s' >$ for $i = 1, \ldots, \ell'$. Since the matrix M' is updated from matrix M, many λ'_i has corresponding value λ_i except the newly added attributes and deleted conflicting attributes. We use re-randomized techniques to reduce the length of the updated ciphertext. So the final ciphertext is as follows: As the third party, the CDMC can get public information $e(g, g)^\alpha$, g to use. He first computes $(e(g,g)^\alpha)^{s'}$ and $g^{s'}$, then multiply them with C_0 and C_1 to get new values. He also computes $W^{\lambda'_i}$ for all $i \in [\ell']$

$$C'_0 = M \cdot e(g, g)^{\alpha \cdot (s+s')}, C'_1 = g^{(s+s')}.$$

If the λ'_i and λ_i correspond to a same attribute, then $W^{\lambda'_i}$ will be multiplied with $C_{i,1}$, and get $C'_{i,1}$.

$$\begin{Bmatrix} C'_{i,1} = W^{\lambda_i + \lambda'_i} V^{t_i}, C'_{i,2} = (U^{x_i} H)^{-t_i}, C'_{i,3} = g^{t_i} & \pi(i) = x_i \\ C'_{i,1} = W^{\lambda_i + \lambda'_i} V'^{t_i}, C'_{i,2} = (U^{x_i} H)^{-t_i}, C'_{i,3} = g^{t_i} & \pi(i) = x'_i \end{Bmatrix}$$

If the λ'_i corresponds to the newly added attributes, then it will choose random value t'_i and computes $C'_{i,1}, C'_{i,2}, C'_{i,3}$ independently.

$$\begin{Bmatrix} C'_{i,1} = W^{\lambda'_i} V^{t'_i}, C'_{i,2} = (U^{x_i} H)^{-t'_i}, C'_{i,3} = g^{t'_i} & \pi(i) = x_i \\ C'_{i,1} = W^{\lambda'_i} V'^{t'_i}, C'_{i,2} = (U^{x_i} H)^{-t'_i}, C'_{i,3} = g^{t'_i} & \pi(i) = x'_i \end{Bmatrix}$$

The final output is $C = (C'_0, C'_1, \{C'_{i,1}, C'_{i,2}, C'_{i,3}\}_{i \in [\ell']})$.

6 Accessing the ciphertext after puncturation. When the ciphertext was punctured by the CDMC, the decryption process is different and can be divided into two subcases:

(1) The decryptor's attribute set ω does not have any revoked attributes. So if the attribute set ω satisfies the original access policy M, it certainly satisfies the newly punctured non-monotonic access policy M'. Then the decryptor can recover two secrets s and s' and recover the message M according to the decryption algorithm in the first case.
(2) The decryptor's attribute set ω contains some revoked attributes. The decryptor may reconstruct the secret s, but the set ω can not satisfy the non-monotonic access policy M',

and the decryptor can not reconstruct the secret s'. Finally, the decryption fails due to having revoked attributes.

4. Security analysis

In this section, the security model is given first. It captures the security aspects of the PABE scheme. Then the security proof of the PABE scheme is given according to the security model.

4.1. Security model

The security of a PABE scheme requires indistinguishability of encrypted message without a valid private key. Let CT be the original ciphertext for access policy $\tilde{\mathbb{A}}$ and CT' be the ciphertext after puncture for access policy $\tilde{\mathbb{A}}'$. The security requires that

- The message in the ciphertext CT cannot be distinguished without a private key whose attribute set satisfies the access policy $\tilde{\mathbb{A}}$.
- The message in the ciphertext CT' cannot be distinguished without a private key whose attribute set satisfies the access policy $\tilde{\mathbb{A}}'$. Most important, the adversary is allowed to have a private key which has a revoked attribute. We define the selective-attributes semantic security for the PABE system. We use one security model to capture two different attacks.

Init: The adversary $\mathcal{A}$ outputs a set of revoked attributes $S^* = \{RA_1^*, \ldots, RA_{s*}^*\}$. The challenger runs $Setup(1^\lambda)$ to obtain the master public key mpk and gives it to the adversary $\mathcal{A}$. **Extraction Query I:** The adversary $\mathcal{A}$ adaptively issues key extraction query for any attribute set ω under the restriction that $\omega \wedge S^* = \phi$. The challenger runs $KeyGen$ on ω and forwards the resulting private key to the adversary. **Challenge:** Once $\mathcal{A}$ decides that **Extraction Query I** is over, it outputs two equal length plaintexts M_0; M_1 and a attribute revocation set AR^*. The only constraint is that any attribute in AR^* cannot have the target attribute in S^*. Let k be the number of attribute in the set AR^*; and S be the set of union of the set S^* and AR^*. The challenger picks a bit $b \in \{0, 1\}$ and generates the challenge ciphertext CT^* as follows:

$$CT = Encrypt(MPK, M_b, \tilde{\mathbb{A}}); CT' = CTPuncture(MPK, ARL, \tilde{\mathbb{A}})$$

The adversary $\mathcal{A}$ is then given the challenge ciphertext $CT^* = CT$ when $ARL \neq \phi$, otherwise it is given $CT^* = CT'$ as the challenge ciphertext. **Extraction Query II :** The adversary $\mathcal{A}$ continues to issue extraction query, as in **Extraction Query I. Guess:** Finally, the adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$. The total number of extraction queries issued by the adversary during the game is denoted by t. We then define the advantage of $\mathcal{A}$ in winning the above game as

$$Adv_{PABE}(t, n, \mathcal{A}) = Pr[b = b'] - 1/2.$$

The probability is over the random coins of $\mathcal{A}$, the challenger and all probabilistic algorithms run by the challenger.

Definition 1. A puncturable attribute based encryption scheme PABE is (t, n) semantically secure if $Adv_{PABE}(t, n) = negl(\lambda)$ for all probabilistic polynomial time adversary $\mathcal{A}$.

4.2. Security proof

Theorem 1. *Suppose the n-Decisional Bilinear Diffie-Hellman Exponent (n-DBDHE) assumption [25] holds. Then no PPT adversary can break the selective security of the PABE scheme with a challenge matrix of size $\ell \times m$ where ℓ, $m \leq n$.*

Since the PABE scheme is an extension of the non-monotonic CP-ABE scheme in [25], so the basic proof strategy is similar. The difference is that the PABE scheme needs simulating the ciphertext puncture algorithm for revoking attributes.

Proof. We construct an algorithm $\mathcal{B}$ which receives difficult problem instance of the n-DBDHE assumption and decides if $T = e(g, g)^{a^{n+1}s}$ using the selective adversary $\mathcal{A}$ against the PABE scheme. □

Init. At the start of the game, the adversary $\mathcal{A}$ declares challenge policy $\tilde{\mathbb{A}}$ where $\tilde{\mathbb{A}} = NM(\mathbb{A}^*)$ and $\mathbb{A}^*$ is specified by (L^*, π^*). L^* is an $\ell \times m$ where ℓ, $m \leq n$. Rows of the matrix is divided into two sets: $positive = \{i | i \in [\ell] \wedge \pi^*(i) = x_i\}$ and $negative = \{i | i \in [\ell] \wedge \pi^*(i) = x'\}$. That is $positive$ and $negative$ is a set of indices that is associated with non-negated and negated attribute. $\mathcal{B}$ chooses random values $\tilde{\alpha}, \tilde{u}, \tilde{v}, \tilde{h} \xleftarrow{s} \mathbb{Z}_p$ and computes:

$$g = g, \quad H = g^{\tilde{h}} \cdot \prod_{(j,k)\in[\ell,m]} \left(g^{a^k/b_j^2}\right)^{-\pi^*(j)L^*_{j,k}}, \quad U = g^{\tilde{u}} \cdot \prod_{(j,k)\in[\ell,m]} \left(g^{a^k/b_j^2}\right)^{L^*_{j,k}}$$

$$W = g^a, \quad V = g^{\tilde{v}} \cdot \prod_{(j,k)\in positive\times[m]} \left(g^{a^k/b_j}\right)^{L^*_{j,k}}, \quad e(g,g)^\alpha = e(g,g)^{\tilde{\alpha}} \cdot e(g^a, g^{a^n}),$$

$\mathcal{B}$ sets $\alpha = \tilde{\alpha} + a^{n+1}$, and sets $b = \sum_{i\in negative} b_i$, and computes:

$$\begin{aligned} V' &= \left(g^{\tilde{u}} \cdot \prod_{(j,k)\in[\ell,m]} \left(g^{a^k/b_j^2}\right)^{L^*_{j,k}}\right)^{\sum_{i\in negative} b_i} \\ &= \left(\prod_{i\in negative} g^{b_i}\right)^{\tilde{u}} \cdot \prod_{i,j,k\in negative\times[\ell,m]} \left(g^{a^k b_i/b_j^2}\right)^{L^*_{j,k}} \\ &= \left(\prod_{i\in negative} g^{b_i}\right)^{\tilde{u}} \cdot \prod_{(i,j,k)\in negative\times[\ell,m], i\neq j} \left(g^{a^k b_i/b_j^2}\right)^{L^*_{j,k}} \cdot \prod_{(j,k)\in negative\times[m]} \left(g^{a^k/b_j}\right)^{L^*_{j,k}}. \end{aligned}$$

Then B gives $MPK = (g, H, U, V, V', W, e(g,g)^\alpha)$ to the adversary $\mathcal{A}$. The value of MPK computed as above is properly distributed. **Extraction Query 1 and 2.** When the adversary $\mathcal{A}$ queries private key for an attribute set $\omega = \{\omega_1, \ldots, \omega_{|\omega|}\}$, $\mathcal{B}$ answers as the following. The process in the two phases is the same. Since $\omega \notin \tilde{\mathbb{A}}^*$, so $\omega' = N(\omega) \notin \mathbb{A}^*$. Therefore $1 = (1, 0, \ldots, 0)$ does not lie in the row space of $L^*_{\omega'}$, which is the sub-matrix of L^* formed by rows corresponding to attributes in ω'. Hence, there must exist an efficiently computable vector $z = (z_1, \ldots, z_m) \in \mathbb{Z}_p^m$ such that $< 1, z >= 1$ and $L_{\omega'} \cdot z^\tau = 0$. $\mathbb{B}$ chooses $\tilde{r} \xleftarrow{s} \mathbb{Z}_p$ and sets

$$\begin{aligned} r &= \tilde{r} - (z_1 a^n + z_2 a^{n-1} + \ldots + z_m a^{n+1-m}) \\ &= \tilde{r} - \sum_{i\in[m]} z_i a^{n+1-i}. \end{aligned}$$

The value of r is properly distributed due to $\tilde{r}$. Then $\mathcal{B}$ can compute:

$$\begin{aligned} D_1 &= g^\alpha W^r = g^{a^{n+1}} g^{\tilde{\alpha}} g^{a\tilde{r}} \prod_{i\in[m]} g^{-z_i a^{n+2-i}} \\ &= g^{\tilde{\alpha}} (g^a)^{\tilde{r}} \prod_{i=2}^{m} (g^{a^{n+2-i}})^{-z_i} D_2 = g^r = g^{\tilde{r}} \prod_{i\in[m]} (g^{a^{n+1-i}})^{-z_i}. \end{aligned}$$

Next, $\mathcal{B}$ computes $K_{i,1}, K_{i,2}, K'_{i,1}, K'_{i,2}$, and gives private key

$$SK_\omega = (D_1, D_2, \{K_{i,1}, K_{i,2}, K'_{i,1}, K'_{i,2}\}_{i\in[|\omega|]})$$

to the adversary $\mathcal{A}$

Challenge. Once the adversary $\mathcal{A}$ decides that one of the key extraction query phases is over, $\mathcal{A}$ gives a pair of plaintext message (M_0, M_1) to $\mathcal{B}$. $\mathcal{B}$'s response to the adversary is divided into two cases.

Case 1: In this case, the attribute revocation list ARL is empty. $\mathcal{B}$ flips a random coin $\beta \xleftarrow{s} \{0, 1\}$ and sets $(C_0, C_1) = (M_\beta \cdot e(g^s, g^{\tilde{\alpha}}) \cdot T, g^s)$ where T is the challenge term and g^s is the corresponding term of the assumption. Next, $\mathcal{B}$ chooses random values $\tilde{s}_2, \ldots, \tilde{s}_m \xleftarrow{s} \mathbb{Z}_p$ and sets $s = (s, sa + \tilde{s}_2), sa^2 + \tilde{s}_3, \ldots, sa^{m-1} + \tilde{s}_m$. So s is properly distributed due to $\{\tilde{s}_i\}_{m=2}^m$. The share λ_τ is computed as follows:

$$\lambda_\tau =< L^*_\tau, s >= \sum_{i\in[m]} L^*_{\tau,i} s a^{i-1} + \sum_{i=2} L^*_{\tau,i} \tilde{s}_i = \sum_{i\in[m]} L^*_{\tau,i} s a^{i-1} + \tilde{\lambda}_\tau,$$

Table 1
Advantage comparison.

	Revocation Level	No Need for Transformation Key	Ciphertext Forward Secrecy	Zero Update for Private Key
[6]	Attribute	No	Yes	No
[12]	Attribute	No	Yes	No
[13]	User	Yes	No	Yes
[15]	User	Yes	No	Yes
[22]	User	Yes	No	Yes
PABE	Attribute	Yes	Yes	Yes

for $\tau \in [\ell]$ where $\tilde{\lambda}_\tau = \sum_{i=2}^{m} L^*_{\tau,i}\tilde{s}_i$. $\mathcal{B}$ then sets $t_\tau = -sb_\tau + \tilde{t}_\tau$ and computes $(C_{\tau,1}, C_{\tau,2}, C_{\tau,3})$ for each $\tau \in [\ell]$. Finally, $\mathcal{B}$ gives the challenge ciphertext $C = (C_0, C_1, C_{i,1}, C_{i,2}, C_{i,3i\in[\ell]})$ to $\mathcal{A}$

Case 2: In this case, the attribute revocation list *ARL* is not empty. The simulation process is different from the first case. The challenger's action is like a real challenger and is called $\mathcal{B}^*$. $\mathcal{B}^*$ first run Encrypt(MPK,M,$\tilde{\mathbb{A}}$) to get ciphertext CT. It picks a random vector $S = (s, s_2, \ldots, s_m) \overset{s}{\leftarrow} \mathbb{Z}_p^m$ and computes share of s for $\pi(i)$ by $\lambda_i =< L_i \cdot s >$ for $i = 1, \ldots, \ell$. It then chooses a random value $t_i =\overset{s}{\leftarrow} \mathbb{Z}_p$ and compute the ciphertext as the follows:

$$C_0 = M \cdot e(g,g)^{\alpha \cdot s}, C_1 = g^s.$$
$$\begin{Bmatrix} C_{i,1} = W^{\lambda_i} V^{t_i}, C_{i,2} = (U^{x_i}H)^{-t_i}, C_{i,3} = g^{t_i} & \pi(i) = x_i \\ C_{i,1} = W^{\lambda_i} V'^{t_i}, C_{i,2} = (U^{x_i}H)^{-t_i}, C_{i,3} = g^{t_i} & \pi(i) = x'_i \end{Bmatrix}$$

$\mathcal{B}^*$ gets the ciphertext as $CT = (C_0, C_1, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i\in[\ell]})$. Then the challenger $\mathcal{B}^*$ runs the algorithm *CTPuncture*(*CT, MPK, ARL*) to revoke attributes. It chooses a new random vector $S' = (s', s'_2, \ldots, s'_m) \overset{\$}{\leftarrow} \mathbb{Z}_p^m$ and computes shares of s' for $\pi'(i)$ by $\lambda'_i =< L_i \cdot S' >$ for $i = 1, \ldots, \ell'$. The challenger $\mathcal{B}^*$ can get public information $e(g, g)^\alpha$, g to use. He first computes $(e(g,g)^\alpha)^{s'}$ and $g^{s'}$, then multiplies them with C_0 and C_1 to get new values. He also computes $W^{\lambda'_i}$ for all $i \in [\ell']$

$$C'_0 = M \cdot e(g,g)^{\alpha \cdot (s+s')}, C'_1 = g^{(s+s')}.$$

If the λ'_i and λ_i correspond to a same attribute, then $W^{\lambda'_i}$ will be multiplied with $C_{i,1}$, and get $C'_{i,1}$.

$$\begin{Bmatrix} C'_{i,1} = W^{\lambda_i+\lambda'_i} V^{t_i}, C'_{i,2} = (U^{x_i}H)^{-t_i}, C'_{i,3} = g^{t_i} & \pi(i) = x_i \\ C'_{i,1} = W^{\lambda_i+\lambda'_i} V'^{t_i}, C'_{i,2} = (U^{x_i}H)^{-t_i}, C'_{i,3} = g^{t_i} & \pi(i) = x'_i \end{Bmatrix}$$

If the λ'_i corresponds to the newly added attributes, then it will choose random value t'_i and computes $C'_{i,1}, C'_{i,2}, C'_{i,3}$ independently.

$$\begin{Bmatrix} C'_{i,1} = W^{\lambda'_i} V^{t'_i}, C'_{i,2} = (U^{x_i}H)^{-t'_i}, C'_{i,3} = g^{t'_i} & \pi(i) = x_i \\ C'_{i,1} = W^{\lambda'_i} V'^{t'_i}, C'_{i,2} = (U^{x_i}H)^{-t'_i}, C'_{i,3} = g^{t'_i} & \pi(i) = x'_i \end{Bmatrix}$$

Finally, the challenger $\mathcal{B}^*$ outputs the punctured ciphertext CT' to the adversary $\mathcal{A}$. $CT' = (C'_0, C'_1, \{C'_{i,1}, C'_{i,2}, C'_{i,3}\}_{i\in[\ell']})$.

Guess. Finally, $\mathcal{A}$ outputs its guess β' for β. If $\beta' = \beta$, $\mathcal{A}$ outputs 1 for its guess. Otherwise, it outputs 0. In the first case, if $T = e(g,g)^{sa^{n+1}}$, the simulated ciphertext is perfect and thus $\mathcal{A}$ has non-negligible advantage. On the other hand, if T is a random element in G_T, $\mathcal{A}$'s advantage is 0. In the second case, the simulated ciphertext returned by the challenger $\mathcal{B}^*$ are actually the same as the ciphertext returned by the challenger $\mathcal{B}$ in the first case when $T = e(g,g)^{sa^{n+1}}$. Therefore, if $\mathcal{A}$ breaks the PABE scheme with non-negligible advantage, $\mathcal{B}$ has a non-negligible advantage in breaking the n-DBDHE assumption.

5. Advantage analysis

The proposed scheme PABE can be used in the cloud computing environment to build secure data access control system with efficient attribute revocation. Compared with other ciphertext policy attribute based encryption schemes supporting attribute revocation, it has three advantages:

(1) No management of ciphertext transformation key. An authorized third party such as the cloud can puncture the ciphertext to revoke attributes using only public information and does not need ciphertext transformation key. This make it greatly different from revocable attribute based encryption schemes based on proxy re-encryption schemes.
(2) Achieving ciphertext forward secrecy. Private keys including revoked attributes immediately lose the decryption capability after the ciphertext puncturation. While in other revocable ABE schemes by adding revocation list during encryption, revoked private keys can decrypt ciphertext generated before revocation.
(3) Zero interaction for private key update. The PABE scheme needs not computing private key update information and interacting with non-revoked users to update their private keys because the ciphertext puncture directly revokes the decryption capability of the revoked attributes.

A detailed comparison between the PABE scheme and other most related schemes is given in Table 1.

6. Conclusion

In this paper, a puncturable attribute based encryption scheme is proposed. It takes a novel way to revoke attributes by puncturing the ciphertext with non-monotonic access policy designed from attribute revocation list. When the non-monotonic access policy is inserted into the ciphertext and merged with the access policy in the ciphertext, any private key including revoked attributes will lose decryption capability for the punctured ciphertext. This scheme has several advantages over traditional methods and is suitable for building flexible secure access control system in the cloud computing environment. ABE schemes supporting non-monotonic access structure is the basis for the construction of PABE scheme. So finding an efficient non-monotonic ABE scheme is the future work and it will greatly improve the efficiency of PABE scheme.

Conflict of Interest

The authors have declared that no conflict of interest exists.

References

[1] Wang Q, He M, Du M, et al. Searchable encryption over feature-rich data. IEEE Trans Dependable Secure Comput 2018;15(3):496–510.
[2] Du M, Wang Q, He M, et al. Privacy-preserving indexing and query processing for secure dynamic cloud storage. IEEE Trans Inf Forensics Secur 2018;13(9):2320–32.
[3] Liu Y, Peng H, Wang J. Verifiable diversity ranking search over encrypted outsourced data. CMC: Comput Mater Continua 2018;55(1):037–57.
[4] Yang Z, Huang Y, Li X, Wang W. Efficient secure data provenance scheme in multimedia outsourcing and sharing. CMC: Comput Mater Continua 2018;56(1):1–17.
[5] Tang Y, Lian H, Zhao Z, Yan X. A proxy re-encryption with keyword search scheme in cloud computing. CMC: Comput Mater Continua 2018;56(2):339–52.

[6] Yu S, Wang C, Ren K, et al. Attribute based data sharing with attribute revocation. In: In Proceedings of the 5th International Symposium on ACM Symposium on Information of Computer and Communications Security; 2010. p. 261–70.
[7] Naruse T, Mohri M, Shiraishi Y. Attribute-based encryption with attribute revocation and grant function using proxy re-encryption and attribute key for updating. In: In Future Information Technology, volume 276 of Lecture Notes in Electrical Engineering. Springer; 2014. p. 119–25.
[8] Ruj S, Nayak A, Stojmenovic I. DACC: Distributed access control in clouds. In: In Proceedings of the TrustCom; 2011. p. 91–8.
[9] Naruse T, Mohri M, Shiraishi Y. Provably secure attribute-based encryption with attribute revocation and grant function using proxy re-encryption and attribute key for updating. Human-centric Comput Inf Sci 2015;5(1):8.
[10] Qian H, Li J, Zhang Y, Han J. Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation. Int J Inf Secur 2015;14(6):487–97.
[11] Hur J, Noh D. Attribute-based access control with efficient revocation in data outsourcing systems. IEEE Trans Parallel Distrib Syst 2011;22(7):1214–21.
[12] Cui H, Deng R, Li Y, et al. Server-aided revocable attribute-based encryption. In: In Proceeding of the 21st European Symposium on Research in Computer Security. Springer; 2016. p. 570–87.
[13] Attrapadung N, Imai H. Conjunctive broadcast and attribute-based encryption. In: In Proceedings of the 3th Pairing-Based Cryptography - Pairing 2009; 2009. p. 248–65. Palo Alto, CA, USA
[14] Attrapadung N, Libert B, de Panafieu E. Expressive key-policy attribute based encryption with constant-size ciphertexts. In: In PKC, volume 6571 of LNCS. Springer; 2011. p. 90–108.
[15] Balu A, Kuppusamy K. Ciphertext-policy attribute-based encryption with user revocation support. In: In QShine, volume 115 of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer; 2013. p. 696–705.
[16] Zhang M. New model and construction of ABE: Achieving key resilient leakage and attribute direct revocation. In: In Proceedings of the 19th Australasian Conference on Information Security and Privacy, volume 8544 of LNCS. Springer; 2014. p. 192–208.
[17] Datta P, Dutta R, Mukhopadhyay S. General circuit realizing compact revocable attribute-based encryption from multilinear maps. In: In ISC, volume 9290 of LNCS. Springer; 2015. p. 336–54.
[18] Liu Z, Wong D. Practical ciphertext-policy attribute-based encryption: Traitor tracing, revocation, and large universe. In: In Proceedings of The 13th International Conference on Applied Cryptography and Network Security, volume 9092 of LNCS. Springer; 2015. p. 127–46.
[19] Datta P, Dutta R, Mukhopadhyay S. Adaptively secure unrestricted attribute-based encryption with subset difference revocation in bilinear groups of prime order. In: Proceedings of the 8th International Conference on Cryptology in Africa, volume 9646 of LNCS. Springer; 2016. p. 325–45.
[20] Liu J, Yuen T, Zhang P, et al. Time-based direct revocable ciphertext-policy attribute-based encryption with short revocation list. In: Proceedings of the International Conference on Applied Cryptography and Network Security. Cham: Springer; 2018. p. 516–34.
[21] Wang H, Zheng Z, Wu L, et al. New directly revocable attribute-based encryption scheme and its application in cloud storage environment. Cluster Comput 2017;20(3):2385–92.
[22] Yang Y, Ding X, Lu H, Wan Z, et al. Achieving revocable fine-grained cryptographic access control over cloud data. In: Proceedings of the 18th Information Security Conference, volume 7807 of LNCS. Springer; 2015. p. 293–308.
[23] Shi J, Huang C, Wang J, He K, Wang J. An access control scheme with direct cloud-aided attribute revocation using version key. In: the Proceedings of the 14th International Conference on Algorithms and Architectures for Parallel Processing Part I, volume 8630 of LNCS. Springer; 2014. p. 429–42.
[24] Yang Y, Liu J, Liang K, et al. Extended proxy-assisted approach: Achieving revocable fine-grained encryption of cloud data. In: 20th European Symposium on Research in Computer Security,Part II, volume 9327 of LNCS. Springer; 2015. p. 146–66.
[25] Yamada S, Attrapadung N, Hanaoka G, et al. A framework and compact constructions for non-monotonic attribute-based encryption. In: Proceedings of the 17th International Conference on Practice and Theory in Public-Key Cryptography; 2014. p. 275–92. Buenos Aires, Argentina
[26] Ostrovsky R, Sahai A, Waters B. Attribute-based encryption with non-monotonic access structures. In: Proceedings of the 2007 ACM Conference on Computer and Communications Security; 2007. p. 195–203. Alexandria, Virginia, USA
[27] Sahai A., Seyalioglu H.. Dynamic credentials and ciphertext delegation for attribute-based encryption. In: International Conference on Advances in Cryptology, CRYPTO 2012. p. 199–217. Santa Barbara, California, USA.
[28] Susilo W, Chen R, Guo F, et al. Recipient revocable identity-based broadcast encryption:how to revoke some recipients in IBBE without knowledge of the plaintext. In: ACM Symposium on Information, Computer and Communications Security, AsiaCCS; 2016. p. 201–10.