Ciphertext Policy Attribute-Based Proxy Re-encryption

Song Luo, Jianbin Hu, and Zhong Chen

Institute of Software, School of Electronics Engineering and Computer Science, Peking University
Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education
{luosong,hjbin,chen}@infosec.pku.edu.cn

Abstract. We present a novel ciphertext policy attribute-based proxy re-encryption (CP-AB-PRE) scheme. The ciphertext policy realized in our scheme is AND-gates policy supporting multi-value attributes, negative attributes and wildcards. Our scheme satisfies the properties of PRE, such as unidirectionality, non-interactivity and multi-use. Moreover, the proposed scheme has master key security, allows the encryptor to decide whether the ciphertext can be re-encrypted and allows the proxy to add access policy when re-encrypting ciphertext. Furthermore, our scheme can be modified to have constant ciphertext size in original encryption.

Keywords: Proxy Re-encryption, Attribute-Based Encryption, Ciphertext Policy.

1 Introduction

A proxy re-encryption (PRE) scheme allows a proxy to translate a ciphertext encrypted under Alice's public key into one that can be decrypted by Bob's secret key. The translation can be performed even by an untrusted proxy. PRE can be used in many scenarios, such as email forwarding: when Alice takes a leave of absence, she can let the others like Bob via the proxy read the message in her encrypted emails. Once Alice comes back, the proxy stops transferring the emails.

Unlike the traditional proxy decryption scheme, PRE does not need users to store any additional decryption key, in other words, any decryption would be finished using only his own secret keys. After Boneh and Franklin [6] proposed a practical identity-base encryption (IBE) scheme, Green and Ateniese [14] proposed the first identity-based PRE (IB-PRE). In IB-PRE schemes, the proxy is allowed to convert an encryption under Alice's identity into the encryption under Bob's identity.

Attribute-based encryption (ABE) is a generalization of IBE. There are two kind of ABE schemes, key policy ABE (KP-ABE) and ciphertext policy ABE (CP-ABE) schemes. In KP-ABE schemes, ciphertexts are associated with sets of attributes and users' secret keys are associated with access policies. In CP-ABE schemes, the situation is reversed. That is, each ciphertext is associated with an access policies. As ABE is the development of IBE, it's natural to implement proxy re-encryption in ABE schemes. However, it is not a trivial work to apply proxy re-encryption technique into attribute based system. ABE uses attributes to identify a group which means some users would have the same attributes. So the ciphertext translation must be extended to many-to-one mapping instead of one-to-one mapping existed in traditional PRE.

Our Contribution. We present a novel ciphertext policy attribute-based proxy re-encryption (CP-AB-PRE) scheme. The ciphertext policy realized in our scheme is AND-gates policy supporting multi-value attributes, negative attributes and wildcards. Our scheme satisfies the following properties of PRE, which are mentioned in [1, 14]:

- *Unidirectionality.* Alice can delegate decryption rights to Bob without permitting her to decrypt Bob's ciphertext.
- *Non-Interactivity.* Alice can compute re-encryption keys without the participation of Bob or the private key generator (PKG).
- *Multi-Use.* The proxy can re-encrypt a ciphertext multiple times, e.g. re-encrypt from Alice to Bob, and then re-encrypt the result from Bob to Carol.

Moreover, our scheme has the other three properties:

- *Master Key Security* [1]. A valid proxy designated by Alice, other users who are able to decrypt Alice's ciphertext with the help from the proxy can not collude to obtain Alice's secret key.
- *Re-encryption Control.* The encryptor can decide whether the ciphertext can be re-encrypted.
- *Extra Access Control.* When the proxy re-encrypts the ciphertext, he can add extra access policy to the ciphertext.

Related Work. Attribute-based encryption was first proposed by Sahai and Waters [24] and later clarified in [13]. Bethencourt, Sahai, and Waters [3] proposed the first CP-ABE scheme. Their scheme allows the ciphertext policies to be very expressive, but the security proof is in the generic group model. Cheung and Newport [9] proposed a provably secure CP-ABE scheme which is proved to be secure under the standard model and their scheme supports AND-Gates policies which deals with negative attributes explicitly and uses wildcards in the ciphertext policies. Goyal et al. [12] proposed a bounded ciphertext policy attribute-based encryption in which a general transformation method was proposed to transform a KP-ABE system into a CP-ABE one by using "universal" access tree. But unfortunately, the parameters of ciphertext and private key sizes will blow up in the worst case. The first secure CP-ABE scheme supporting general access formulas was presented by Waters [25]. By using the dual system encryption techniques [26, 17], Lewko et al. [18] present a fully secure CP-ABE scheme.

There are many other ABE schemes. Multiple authorities were introduced in [7] and [8]. K.Emura et al. [11] introduced a novel scheme using AND-Gates policy which has constant ciphertext length. T.Nishide et al. [22] gave a method on how to hide access structure in attribute-based encryption. R.Bobba et al. [5] enhanced attribute-based encryption with attribute-sets which allow same attributes in different sets. N.Attrapadung et al. [2] proposed dual-policy attribute-based encryption which allows simultaneously key-policy and ciphertext-policy act on encrypted data. Recently, a generalization of ABE called predicate (or functional) encryption was proposed by Katz, Sahai, and Waters [16] and furthered by T.Okamoto et al. [23].

The notion of PRE was first introduced by Mambo and Okamoto [20]. Later Blaze et al. [4] proposed the first concrete bidirectional PRE scheme which allows the keyholder to publish the proxy function and have it applied by untrusted parties without further involvement by the original keyholder. Their scheme also has multi-use property. Ateniese et al. [1] presented the first unidirectional and single-use proxy re-encryption scheme. In 2007, Green and Ateniese [14] provided identity-based PRE but their schemes are secure in the random oracle model. Chu et al. [10] proposed new identity-based proxy re-encryption schemes in the standard model. Matsuo [21] also proposed new proxy re-encryption system for identity-based encryption, but his solution needs a re-encryption key generator (RKG) to generate re-encryption keys. After the present of ABE, Guo et al. [15] proposed the first attribute-based proxy re-encryption scheme, but their scheme is based on key policy and bidirectional. Liang et al. [19] proposed the first ciphertext policy attribute-based proxy re-encryption scheme which has the above properties except re-encryption control.

Organization. The paper is organized as follows. We give necessary background information and our definitions of security in Section 2. We then present our construction and give a proof of security in Section 3 and discuss a number of extensions of the proposed scheme in Section 4. Finally, we conclude the paper with Section 5.

2 Preliminaries

2.1 Bilinear Maps and Complexity Assumptions

Definition 1 (Bilinear Maps)
Let $\mathbb{G}$, $\mathbb{G}_1$ be two cyclic multiplicative groups with prime order p. Let g be be a generator of $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ be a bilinear map with the following properties:

1. Bilinearity: $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

2. Non-degeneracy: The map does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_1$. Observe that since $\mathbb{G}, \mathbb{G}_1$ are groups of prime order this implies that if g is a generator of $\mathbb{G}$ then $e(g, g)$ is a generator of $\mathbb{G}_1$.

We say that $\mathbb{G}$ is a bilinear group if the group operation in $\mathbb{G}$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ are both efficiently computable.

We assume that there is an efficient algorithm *Gen* for generating bilinear groups. The algorithm *Gen*, on input a security parameter κ, outputs a tuple $G = [p, \mathbb{G}, \mathbb{G}_1, g \in \mathbb{G}, e]$ where $\log(p) = \Theta(\kappa)$.

We describe the Computational Bilinear Diffie-Hellman (CBDH) problem and the Decisional Bilinear Diffie-Hellman (DBDH) assumption used in our security proofs.

Definition 2 (CBDH Problem)
Let $a, b, c \in_R \mathbb{Z}$ and $g \in \mathbb{G}$ be a generator. The CBDH problem is given the tuple $[g, g^a, g^b, g^c]$, to output $e(g,g)^{abc}$.

Definition 3 (DBDH Assumption)
Let $a, b, c, z \in_R \mathbb{Z}$ and $g \in \mathbb{G}$ be a generator. The DBDH assumption is that no probabilistic polynomial-time algorithm can distinguish the tuple $[g, g^a, g^b, g^c, e(g,g)^{abc}]$ from the tuple $[g, g^a, g^b, g^c, e(g,g)^z]$ with non-negligible advantage.

2.2 Access Policy

In ciphertext policy attribute-based encryption, access policy is associated with ciphertext specifying who can decrypt the ciphertext. In our scheme, we use AND-gates on multi-valued attributes, negative attributes and wildcards. Negative attribute is used to specify that the user doesn't have this attribute. Wildcard means the attribute does not need consideration in decryption.

Definition 4. *Let $\mathcal{U} = \{att_1, \cdots, att_n\}$ be a set of attributes. For each $att_i \in \mathcal{U}$, $S_i = \{v_{i,1}, \cdots, v_{i,n_i}\}$ be a set of possible values, where $n_i = |S_i|$ is the number of possible values for att_i. Let $\bar{\mathcal{U}} = \{\neg att_1, \cdots, \neg att_n\}$ a set of negative attributes for $\mathcal{U}$. Let $L = [L_1, \cdots, L_n], L_i \in S_i \cup \{\neg att_i\}$ be an attribute list for a user; and $W = [W_1, \cdots, W_n], W_i \in S_i \cup \{\neg att_i, *\}$ be an access policy.*

The notation $L \models W$ means that an attribute list L satisfies an access policy W, namely, for all $i = 1, \cdots, n$, $L_i = W_i$ or $W_i = *$. Otherwise, we use notation $L \not\models W$ to mean L does not satisfy W.

2.3 Algorithms of CP-AB-PRE

A CP-AB-PRE scheme consists of the following six algorithms: **Setup**, **KeyGen**, **Encrypt**, **RKGen**, **Reencrypt**, and **Decrypt**.
Setup(1^κ). This algorithm takes the security parameter κ as input and generates a public key PK , a master secret key MK.
KeyGen(MK, L). This algorithm takes MK and a set of attributes L as input and generates a secret key SK_L associated with L.
Encrypt(PK, M, W). This algorithm takes PK, a message M, and an access policy W as input, and generates a ciphertext CT_W.
RKGen(SK_L, W). This algorithm takes a secret key SK_L and an access policy W as input and generates a re-encryption key $RK_{L \to W}$.
Reencrypt$(RK_{L\to W'}, CT_W)$. This algorithm takes a re-encryption key $RK_{L\to W'}$ and a ciphertext CT_W as input, first checks if the attribute list in $RK_{L\to W'}$ satisfies the access policy of CT_W, that is, $L \overset{?}{\models} W$. Then, if check passes, it generates a re-encrypted ciphertext $CT_{W'}$; otherwise, it returns $\perp$.

Decrypt(CT_W, SK_L). This algorithm takes CT_W and SK_L associated with L as input and returns the message M if the attribute list L satisfies the access policy W specified for CT_W, that is, $L \overset{?}{\models} W$. If $L \not\models W$, it returns $\perp$ with overwhelming probability.

2.4 Security Model

We describe the security model called Selective-Policy Model for our CP-AB-PRE scheme. Based on [19], we use the following security game. A CP-AB-PRE scheme is selective-policy chosen plaintext secure if no probabilistic polynomial-time adversary has non-negligible advantage in the following Selective-Policy Game.

Selective-Policy Game for CP-AB-PRE
Init: The adversary $\mathcal{A}$ commits to the challenge ciphertext policy W^*.
Setup: The challenger runs the **Setup** algorithm and gives PK to $\mathcal{A}$.
Phase 1: $\mathcal{A}$ makes the following queries.

- **Extract**(L): $\mathcal{A}$ submits an attribute list L for a **KeyGen** query where $L \not\models W^*$, the challenger gives the adversary the secret key SK_L.
- **RKExtract**(L, W): $\mathcal{A}$ submits an attribute list L for a **RKGen** query where $L \not\models W^*$, the challenger gives the adversary the re-encryption key $SK_{L \to W}$.

Challenge: $\mathcal{A}$ submits two equal-length messages M_0, M_1 to the challenger. The challenger flips a random coin μ and passes the ciphertext **Encrypt**(PK, M_μ, W^*) to the adversary.
Phase 2: Phase 1 is repeated.
Guess: $\mathcal{A}$ outputs a guess μ' of μ.
The advantage of $\mathcal{A}$ in this game is defined as $Adv_{\mathcal{A}} = |Pr[\mu' = \mu] - \frac{1}{2}|$ where the probability is taken over the random bits used by the challenger and the adversary.

In [1], Ateniese et al. defined another important security notion, named delegator secret security (or master key security), for unidirectional PRE. This security notion captures the intuition that, even if the dishonest proxy colludes with the delegatee, it is still impossible for them to derive the delegator's private key in full.

We give master key security game for attribute-based proxy re-encryption as follows. A CP-AB-PRE scheme has selective master key security if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in winning the following selective master key security game.

Selective Master Key Security Game
Init: The adversary $\mathcal{A}$ commits to a challenge attribute list L^*.
Setup: The challenger runs the **Setup** algorithm and gives PK to $\mathcal{A}$.
Queries: $\mathcal{A}$ makes the following queries.

- Extract(L): $\mathcal{A}$ submits an attribute list L for a **KeyGen** query where $L \neq L^*$, the challenger gives the adversary the secret key SK_L.

– RKExtract(L, W): $\mathcal{A}$ submits an attribute list L for a **RKGen** query, the challenger gives the adversary the re-encryption key $SK_{L \to W}$.

Output: $\mathcal{A}$ outputs the secret key SK_{L^*} for the attribute list L^*.
The advantage of $\mathcal{A}$ in this game is defined as $Adv_{\mathcal{A}} = Pr[\mathcal{A}$ succeeds$]$.

3 Proposed Scheme

3.1 Our Construction

Let $\mathbb{G}$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}$. In addition, let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$ denote the bilinear map. Let $E : \mathbb{G} \to \mathbb{G}_1$ be an encoding between $\mathbb{G}$ and $\mathbb{G}_1$. A security parameter, κ, will determine the size of the groups. Let $\mathcal{U} = \{att_1, \cdots, att_n\}$ be a set of attributes; $S_i = \{v_{i,1}, \cdots, v_{i,n_i}\}$ be a set of possible values associated with att_i and $n_i = |S_i|$; $L = [L_1, \cdots, L_n]$ be an attribute list for a user; and $W = [W_1, \cdots, W_n]$ be an access policy.

Our six algorithms are as follows:
Setup(1^κ). A trusted authority (TA) generates a tuple $G = [p, \mathbb{G}, \mathbb{G}_1, g \in \mathbb{G}, e] \leftarrow Gen(1^\kappa)$, $y \in_R \mathbb{Z}_p$ and $g_2, g_3 \in_R \mathbb{G}$. For each attribute att_i where $1 \leqslant i \leqslant n$, TA generates values $\{t_{i,j} \in_R \mathbb{Z}_p\}_{1 \leqslant j \leqslant n_i}$ and $\{a_i, b_i \in_R \mathbb{Z}_p\}$. Next TA computes $g_1 = g^y, Y = e(g_1, g_2), \{\{T_{i,j} = g^{t_{i,j}}\}_{1 \leqslant j \leqslant n_i}, A_i = g^{a_i}, B_i = g^{b_i}\}_{1 \leqslant i \leqslant n}$. The public key PK is published as

$$PK = \langle e, g, g_1, g_2, g_3, Y, \{\{T_{i,j}\}_{1 \leqslant j \leqslant n_i}, A_i, B_i\}_{1 \leqslant i \leqslant n} \rangle,$$

The master key MK is $MK = \langle y, \{\{t_{i,j}\}_{1 \leqslant j \leqslant n_i}, a_i, b_i\}_{1 \leqslant i \leqslant n} \rangle$.
KeyGen(MK, L). Let $L = [L_1, L_2, \cdots, L_n]$ be the attribute list for the user who obtains the corresponding secret key. TA chooses $r_i, r'_i, r''_i \in_R \mathbb{Z}_p$ for $1 \leqslant i \leqslant n$, set $r = \sum_{i=1}^n r_i$ and computes $D_0 = g_2^{y-r}$. TA computes D_i and F_i for $1 \leqslant i \leqslant n$ as

$$D_i = \begin{cases} (g_2^{r_i} T_{i,k_i}^{r'_i},\ g^{r'_i}) \text{ (if } L_i = v_{i,k_i}) \\ (g_2^{r_i} A_i^{r'_i},\ g^{r'_i}) \ \text{ (if } L_i = \neg att_i) \end{cases}, F_i = (g_2^{r_i} B_i^{r''_i},\ g^{r''_i}).$$

TA outputs the secret key $SK_L = \langle L, D_0, \{D_i, F_i\}_{1 \leqslant i \leqslant n} \rangle$.
Encrypt(PK, M, W). To encrypt a message $M \in \mathbb{G}_1$ under the access policy W, an encryptor chooses $s \in_R \mathbb{Z}_p$, computes $\tilde{C} = M \cdot Y^s$ and $C_0 = g^s, C'_0 = g_3^s$. Then the encryptor computes C_i for $1 \leqslant i \leqslant n$ as follows:

$$C_i = \begin{cases} T_{i,k_i}^s & \text{(if } W_i = v_{i,k_i}) \\ A_i^s & \text{(if } W_i = \neg att_i) \\ B_i^s & \text{(if } W_i = *) \end{cases}$$

The encryptor outputs the ciphertext $CT_W = \langle W, \tilde{C}, C_0, C'_0, \{C_i\}_{1 \leqslant i \leqslant n} \rangle$.
RKGen(SK_L, W). Let SK_L denote a valid secret key, W an access policy. To generate a re-encryption key for W, chooses $d \in_R \mathbb{Z}_p$ and computes $g^d, D'_i = (D_{i,1} g_3^d,\ D_{i,2}), F'_i = (F_{i,1} g_3^d,\ F_{i,2})$. Sets $D'_0 = D_0$ and computes $\mathbb{C}$ which is the

ciphertext of $E(g^d)$ under the access policy W, i.e., $\mathbb{C} = \textbf{Encrypt}(PK, E(g^d), W)$. Then the re-encryption key for W is $RK_{L\to W} = \langle L, W, D'_0, \{D'_i, F'_i\}_{1\leqslant i\leqslant n}, \mathbb{C}\rangle$.
Reencrypt$(RK_{L\to W'}, CT_W)$. Let $RK_{L\to W'}$ be a valid re-encryption key for access policy W', CT_W a well-formed ciphertext $\langle W, \tilde{C}, C_0, C'_0, \{C_i\}_{1\leqslant i\leqslant n}\rangle$, checks W to know whether $L \models W$. If $L \not\models W$, returns $\perp$; otherwise, for $1 \leqslant i \leqslant n$, computes:

$$E_i = \left\{ \begin{array}{l} \frac{e(C_0, D'_{i,1})}{e(C_i, D'_{i,2})} \text{ (if } W'_i \neq *) \\ \frac{e(C_0, F'_{i,1})}{e(C_i, F'_{i,2})} \text{ (if } W'_i = *) \end{array} \right\} = e(g, g_2)^{sr_i} e(g, g_3)^{sd}.$$

It then computes $\bar{C} = e(C_0, D'_0) \prod_{i=1}^n E_i = e(g^s, g_2^{y-r}) e(g, g_2)^{sr} e(g, g_3)^{nsd} = e(g, g_2)^{ys} e(g, g_3)^{nsd}$, outputs a re-encrypted ciphertext

$$CT' = \langle W', \tilde{C}, C'_0, \bar{C}, \mathbb{C}\rangle.$$

Note that $\mathbb{C}$ can be re-encrypted again. In the following **Decrypt** algorithm we can see that the recipient only needs g^d to decrypt the re-encrypted ciphertext. Thus, we would obtain $CT'' = \langle W'', \tilde{C}, C'_0, \bar{C}, \mathbb{C}'\rangle$, where $\mathbb{C}'$ is obtained from the **Reencrypt** algorithm with the input of another $RK_{L'\to W''}$ and $\mathbb{C}$. The decryption cost and size of ciphertext grows linearly with the re-encryption times. As stated in [14], it seems to be inevitable for a non-interactive scheme.
Decrypt(CT_W, SK_L). A decryptor checks W to know whether $L \models W$. If $L \models W$, she can proceed. Then the decryptor decrypts the CT by using her SK_L as follows:

– If CT is an original well-formed ciphertext, then
  1. For $1 \leqslant i \leqslant n$, computes

$$D'_i = \begin{cases} D_{i,1} \text{ (if } W_i \neq *) \\ F_{i,1} \text{ (if } W_i = *) \end{cases}, D''_i = \begin{cases} D_{i,2} \text{ (if } W_i \neq *) \\ F_{i,2} \text{ (if } W_i = *) \end{cases},$$

  2. $M = \tilde{C} \prod_{i=1}^n e(C_i, D''_i) / (e(C_0, D_0) \prod_{i=1}^n e(C_0, D'_i))$.
– Else if CT is a re-encrypted well-formed ciphertext, then
  1. Decrypts $E(g^d)$ from $\mathbb{C}$ using the secret key SK_L and decodes it to g^d,
  2. $M = \tilde{C} \cdot e(C'_0, g^d)^n / \bar{C}$.
– Else if CT is a multi-time re-encrypted well-formed ciphertext, decryption is similar with the above phases.

Correctness. The correctness is easily observable.

3.2 Security Proof

Theorem 1. *If there is an adversary who breaks our scheme in the Selective-Policy model, a simulator can take the adversary as oracle and break the DBDH assumption with a non-negligible advantage.*

Proof. We will show that a simulator $\mathcal{B}$ can break the DBDH assumption with advantage $\epsilon/2$ if it takes an adversary $\mathcal{A}$, who can break our scheme in the Selective-Set model with advantage ϵ, as oracle.

Given a DBDH challenge $[g, A, B, C, Z] = [g, g^a, g^b, g^c, Z]$ by the challenger where Z is either $e(g,g)^{abc}$ or random with equal probability, the simulator $\mathcal{B}$ creates the following simulation.

Init: The simulator $\mathcal{B}$ runs $\mathcal{A}$. $\mathcal{A}$ gives $\mathcal{B}$ a challenge ciphertext policy W^*.

Setup: To provide a public key PK to $\mathcal{A}$, $\mathcal{B}$ sets $g_1 = A, g_2 = B$ and computes $Y = e(A,B) = e(g,g)^{ab}$. $\mathcal{B}$ chooses $\gamma \in_R \mathbb{Z}_p$ and computes $g_3 = g^{\gamma}$. $\mathcal{B}$ chooses $S_i = \{t_{i,j}\}_{1 \leqslant j \leqslant n_i}, a_i, b_i \in_R \mathbb{Z}_p$ and constructs $T_{i,j}, A_i, B_i$ as follows:

- If $W_i^* = v_{i,k_i}$, then $T_{i,j} = \begin{cases} g^{t_{i,j}} & (j = k_i) \\ B^{t_{i,j}} & (j \neq k_i) \end{cases}, A_i = B^{a_i}, B_i = B^{b_i}$;
- If $W_i^* = \neg att_i$, then $T_{i,j} = B^{t_{i,j}}, A_i = g^{a_i}, B_i = B^{b_i}$;
- If $W_i^* = *$, then $T_{i,j} = B^{t_{i,j}}, A_i = B^{a_i}, B_i = g^{b_i}$.

Finally $\mathcal{B}$ sends $\mathcal{A}$ the public key.

Phase 1: $\mathcal{A}$ makes the following queries.

- **Extract**(L): $\mathcal{A}$ submits an attribute list $L = (L_1, L_2, \cdots, L_n)$ in a secret key query. The attribute list must satisfying $L \not\models W^*$ or else $\mathcal{B}$ simply aborts and takes a random guess.

  Since $L \not\models W^*$, there must exist some j such that: either W_j^* is an attribute value and $L_j \neq W_j^*$,or W_j^* is a negative attribute $\neg att_i$ and $L_j \in S_j$. $\mathcal{B}$ chooses such j. Without loss of generality, assume that W_j^* is an attribute value and $L_j \neq W_j^*$.

  For $1 \leqslant i \leqslant n$, $\mathcal{B}$ chooses $r_i' \in_R \mathbb{Z}_p$. It then sets r_i as follows:

$$r_i = \begin{cases} r_i' & (i \neq j) \\ a + r_i' & (i = j) \end{cases}$$

  Finally, it sets $r := \sum_{i=1}^{n} r_i = a + \sum_{i=1}^{n} r_i'$. The D_0 component of the secret key can be computed as $\prod_{i=1}^{n} B^{-r_i'} = g_2^{-\sum_{i=1}^{n} r_i'} = g_2^{a-r}$.

  Recall that W_j is an attribute value and $L_j \neq W_j^*$, let $L_j = v_{j,k}$, $\mathcal{B}$ chooses $\beta \in_R \mathbb{Z}_p$ and sets $t = \frac{\beta - a}{t_{j,k}}$, then computes component D_j as

$$D_j = (B^{\beta + r_j'},\ g^{\frac{\beta}{t_{j,k}}} A^{-\frac{1}{t_{j,k}}}) = (g_2^{r_j} T_{j,k}^{t},\ g^{t}).$$

  For $i \neq j$, $\mathcal{B}$ chooses $t_i \in_R \mathbb{Z}_p$ and computes component D_i as follows:
  - If $L_i = v_{i,k_i}$ is an attribute value, then $D_i = (B^{r_i} T_{i,k_i}^{t_i},\ g^{t_i})$;
  - If $L_i = \neg att_i$ is a negative attribute, then $D_i = (B^{r_i} {A_i}^{t_i},\ g^{t_i})$.

  The F_i components are computed similarly. $\mathcal{B}$ chooses $\delta \in_R \mathbb{Z}_p$ and sets $t' = \frac{\delta - a}{b_j}$, then computes component F_j as

$$F_j = (B^{\delta + r_j'},\ g^{\frac{\delta}{b_j}} A^{-\frac{1}{b_j}}) = (g_2^{r_j} B_j^{t'},\ g^{t'})$$

  For $i \neq j$, $\mathcal{B}$ chooses $t_i' \in_R \mathbb{Z}_p$ and computes component F_i as follows:

$$F_i = (B^{r_i} {B_i}^{t_i'},\ g^{t_i'}).$$

– **RKExtract**(L, W): $\mathcal{A}$ submits an attribute list $L = (L_1, L_2, \cdots, L_n)$ and an access policy $W = (W_1, W_2, \cdots, W_n)$ in a re-encryption key query. The attribute list must satisfying $L \not\models W^*$ or else $\mathcal{B}$ simply aborts and takes a random guess.
  Then $\mathcal{B}$ submits L to **Extract** query and gets a secret key $SK_L = \langle L, D_0, \{D_i, F_i\}_{1 \leqslant i \leqslant n}\rangle$. To generate a re-encryption key for W, chooses $d \in_R \mathbb{Z}_p$ and computes $g^d, D'_i = (D_{i,1}g_3^d,\ D_{i,2}), F'_i = (F_{i,1}g_3^d,\ F_{i,2})$. Sets $D'_0 = D_0$ and computes $\mathbb{C}$ which is the ciphertext of $E(g^d)$ under the access policy W. Then the re-encryption key for W is

$$RK_{L \to W} = \langle L, W, D'_0, \{D'_i, F'_i\}_{1 \leqslant i \leqslant n}, \mathbb{C}\rangle.$$

Challenge: $\mathcal{A}$ submits two challenge messages M_0 and M_1. $\mathcal{B}$ sets $C_0 = C$, computes $C'_0 = g_3^s = (g^\gamma)^s = (g^s)^\gamma = C^\gamma$ and C_i for $1 \leqslant i \leqslant n$ as follows:

– If $W_i = v_{i,k_i}$, then $C_i = C^{t_{i,k_i}}$;
– Else if $W_i = \neg att_i$, then $C_i = C^{a_i}$;
– Else if $W_i = *$, then $C_i = C^{b_i}$.

Then $\mathcal{B}$ flips a random coin $\mu \in \{0, 1\}$ and returns $\mathcal{A}$ the ciphertext as $\langle \tilde{C} = M_\mu Z, C_0, C'_0, \{C_i\}_{1 \leqslant i \leqslant n}\rangle$.
Phase 2: Phase 1 is repeated.
Guess: $\mathcal{A}$ outputs a guess μ' of μ. $\mathcal{B}$ outputs 1 if and only if $\mu' = \mu$.
Therefore, the advantage of breaking the DBDH assumption is

$$\begin{aligned} Adv_{\mathcal{A}} &= |Pr[\mu' = \mu] - \frac{1}{2}| \\ &= |Pr[\mu = 0]Pr[\mu' = \mu|\mu = 0] + Pr[\mu = 0]Pr[\mu' = \mu|\mu = 1] - \frac{1}{2}| \\ &= |\frac{1}{2}(\frac{1}{2} + \epsilon) + \frac{1}{2}\frac{1}{2} - \frac{1}{2}| \\ &= \frac{1}{2}\epsilon \end{aligned}$$

□

Theorem 2. *If there is an adversary who breaks our scheme in selective the master key security model, a simulator can take the adversary as oracle and solve the CBDH problem with a non-negligible advantage.*

Proof. We will show that a simulator $\mathcal{B}$ can solve the CBDH problem with advantage ϵ if it takes an adversary $\mathcal{A}$, who can break our scheme in the selective master key security model with advantage ϵ, as oracle.

Given a CBDH challenge tuple $[g, A, B, C] = [g, g^a, g^b, g^c]$ by the challenger, the simulator $\mathcal{B}$ creates the following simulation.
Init: The simulator $\mathcal{B}$ runs $\mathcal{A}$. $\mathcal{A}$ gives $\mathcal{B}$ a challenge attribute list L^*.
Setup: To provide a public key PK to $\mathcal{A}$, $\mathcal{B}$ sets $g_1 = A, g_2 = B, g_3 = B$ and computes $Y = e(A, B) = e(g, g)^{ab}$. $\mathcal{B}$ chooses $S_i = \{t_{i,j}\}_{1 \leqslant j \leqslant n_i}, a_i, b_i \in_R \mathbb{Z}_p$ and constructs $T_{i,j}, A_i, B_i$ as follows:

- If $L_i^* = v_{i,k}$, then $T_{i,j} = \begin{cases} g^{t_{i,j}} & (j = k) \\ B^{t_{i,j}} & (j \neq k) \end{cases}$, $A_i = B^{a_i}, B_i = B^{b_i}$;
- If $L_i^* = \neg att_i$, then $T_{i,j} = B^{t_{i,j}}, A_i = g^{a_i}, B_i = B^{b_i}$.

Then $\mathcal{B}$ sends $\mathcal{A}$ the public key.
Queries: $\mathcal{A}$ makes the following queries.

- **Extract**(L): $\mathcal{A}$ submits an attribute list $L = (L_1, L_2, \cdots, L_n)$ in a secret key query. The attribute list must satisfying $L \neq L^*$ or else $\mathcal{B}$ simply aborts and takes a random guess.
  Since $L \neq L^*$, there must exist some j such that $L_j \neq L_j^*$. $\mathcal{B}$ chooses such j. For $1 \leqslant i \leqslant n$, $\mathcal{B}$ chooses $r_i' \in_R \mathbb{Z}_p$. It then sets r_i as follows:

$$r_i = \begin{cases} r_i' & (i \neq j) \\ a + r_i' & (i = j) \end{cases}$$

  Finally, it sets $r := \sum_{i=1}^n r_i = a + \sum_{i=1}^n r_i'$. The D_0 component of the secret key can be computed as $\prod_{i=1}^n B^{-r_i'} = g_2^{-\sum_{i=1}^n r_i'} = g_2^{a-r}$.
  Recall that $L_j \neq L_j^*$, let $L_j = v_{j,k}$, $\mathcal{B}$ chooses $\beta \in_R \mathbb{Z}_p$ and sets $t = \frac{\beta - a}{t_{j,k}}$, then computes $D_j = (B^{\beta + r_j'},\ g^{\frac{\beta}{t_{j,k}}} A^{-\frac{1}{t_{j,k}}}) = (g_2^{r_j} T_{j,k}^t,\ g^t)$.
  For $i \neq j$, $\mathcal{B}$ chooses $t_i \in_R \mathbb{Z}_p$ and computes component D_i as follows:
  - If $L_i = v_{i,k}$ is an attribute value, then $D_i = (B^{r_i} T_{i,k}^{t_i},\ g^{t_i})$;
  - If $L_i = \neg att_i$ is a negative attribute, then $D_i = (B^{r_i} A_i^{t_i},\ g^{t_i})$.

  The F_i components are computed similarly. $\mathcal{B}$ chooses $\delta \in_R \mathbb{Z}_p$ and sets $t' = \frac{\delta - a}{b_j}$, then computes component F_j as

$$F_j = (B^{\delta + r_j'},\ g^{\frac{\delta}{b_j}} A^{-\frac{1}{b_j}}) = (g_2^{r_j} B_j^{t'},\ g^{t'})$$

  For $i \neq j$, $\mathcal{B}$ chooses $t_i' \in_R \mathbb{Z}_p$ and computes component F_i as follows:

$$F_i = (B^{r_i} B_i^{t_i'},\ g^{t_i'}).$$

- **RKExtract**(L, W): $\mathcal{A}$ submits an attribute list $L = (L_1, L_2, \cdots, L_n)$ and an access policy $W = (W_1, W_2, \cdots, W_n)$ in a re-encryption key query.
  If $L \neq L^*$, $\mathcal{B}$ submits L to **Extract** query and gets a secret key $SK_L = \langle L, D_0, \{D_i, F_i\}_{1 \leqslant i \leqslant n} \rangle$. To generate a re-encryption key for W, chooses $d \in_R \mathbb{Z}_p$ and computes $g^d, D_i' = (D_{i,1} g_3^d,\ D_{i,2}), F_i' = (F_{i,1} g_3^d,\ F_{i,2})$. Sets $D_0' = D_0$ and computes $\mathbb{C}$ which is the ciphertext of $E(g^d)$ under the access policy W.
  If $L = L^*$, $\mathcal{B}$ computes the corresponding re-encryption key as follows. First, for $1 \leqslant i \leqslant n$, $\mathcal{B}$ chooses $r_i' \in_R \mathbb{Z}_p$ and sets $r_i = \frac{a}{n} + r_i'$. Then it sets $r := \sum_{i=1}^n r_i = a + \sum_{i=1}^n r_i'$. The D_0' component of the re-encryption key can be computed as $\prod_{i=1}^n B^{-r_i'} = g_2^{-\sum_{i=1}^n r_i'} = g_2^{a-r}$. Next, chooses $d' \in_R \mathbb{Z}_p$ and sets $d = -\frac{a}{n} + d'$ which means $g^d = g^{-\frac{a}{n} + d'} = A^{-\frac{1}{n}} g^{d'}$. For $1 \leqslant i \leqslant n$, $\mathcal{B}$ chooses $t_i \in_R \mathbb{Z}_p$ and computes component D_i' as follows:

- If $L_i = v_{i,k_i}$ is an attribute value, then

$$D'_i = (g_2^{r_i} T_{i,k_i}^{t_i} g_3^d,\ g^{t_i}) = (B^{r_i} T_{i,k_i}^{t_i} B^d,\ g^{t_i}) = (B^{r'_i+d'} T_{i,k_i}^{t_i},\ g^{t_i})$$

- If $L_i = \neg att_i$ is a negative attribute, then

$$D'_i = (g_2^{r_i} A_i^{t_i} g_3^d,\ g^{t_i}) = (B^{r_i} A_i^{t_i} B^d,\ g^{t_i}) = (B^{r'_i+d'} A_i^{t_i},\ g^{t_i})$$

The F_i components are computed similarly. Finally, computes $\mathbb{C}$ which is the ciphertext of $E(g^d)$ under the access policy W.
Then the re-encryption key is $RK_{L\to W} = \langle L, W, D'_0, \{D'_i, F'_i\}_{1\leqslant i\leqslant n}, \mathbb{C}\rangle$.

Output: $\mathcal{A}$ outputs the secret key $SK_{L^*} = \langle L^*, D_0, \{D_i, F_i\}_{1\leqslant i\leqslant n}\rangle$ for the attribute list L^*.

If it is a valid secret key, SK_{L^*} should satisfy the following equation:

$$e(g^s, D_0)\prod_{i=1}^{n}\frac{e(g^s, D_{i,1})}{e(X_i, D_{i,2})} = e(g_1, g_2)^s,$$

where $s \in_R \mathbb{Z}_p$ and $X_i = T_{i,k}^s$ (if $L_i^* = v_{i,k}$) or $X_i = A_i^s$ (if $L_i^* = \neg att_i$).

Thus, $\mathcal{B}$ outputs $e(C, D_0)\prod_{i=1}^{n}\frac{e(C,D_{i,1})}{e(C_i,D_{i,2})} = e(A,B)^c = e(g,g)^{abc}$ where $C_i = C^{t_{i,k}}$ (if $L_i^* = v_{i,k}$) or $C_i = C^{a_i}$ (if $L_i^* = \neg att_i$) and solves the CBDH problem.

By the simulation setup the simulator's advantage in solving the CBDH problem is exactly ϵ. □

4 Discussions

4.1 Computation Reduction

The number of pairings in decrypting original ciphertext is $2n+1$. Observe that in pairing $e(C_0, D'_i)$, C_0 is constant so the multiple multiplication $\prod_{i=1}^{n} e(C_0, D'_i)$ (we can also add $e(C_0, D_0)$) can be replaced with $e(C_0, D_0\prod_{i=1}^{n} D'_i)$ which may reduce the number of pairings to n.

The same reduction can be done in re-encryption algorithm. Because every E_i needs compute $e(C_0, D'_{i,1})$ or $e(C_0, F'_{i,1})$ and multiplies together, we can also precompute $\prod_{i=0}^{n} e(C_0, X)$ where $X = D'_0$ or $X = D'_{i,1}$ or $X = F'_{i,1}$.

There is still room for improvement. Observe that in pairing $e(C_i, D''_i)$, $D''_i = g^{r'_i}$ or $D''_i = g^{r''_i}$ is a random element for every attribute. If D''_i is constant, that is, all $D_{i,2} = g^{r'}$. Then the multiple multiplication $\prod_{i=1}^{t} e(C_i, D''_i)$ can be transferred to $e(\prod_{i=1}^{t} C_i, g^{r'})$ which may reduce pairings and make the original ciphertext size be constant.

To achieve this goal, we must make some modifications. We remove wildcard in the policy to avert choosing the corresponding component $D_{i,2}$ or $F_{i,2}$. The full modified scheme is as follows.

Setup(1^κ). A trusted authority (TA) generates a tuple $G = [p, \mathbb{G}, \mathbb{G}_1, g \in \mathbb{G}, e] \leftarrow Gen(1^\kappa)$, $y \in_R \mathbb{Z}_p$ and $g_2, g_3 \in_R \mathbb{G}$. For each attribute att_i where $1 \leqslant i \leqslant n$, TA generates values $\{t_{i,j} \in_R \mathbb{Z}_p\}_{1\leqslant j\leqslant n_i}$. Next TA computes

$$g_1 = g^y, Y = e(g_1, g_2), \{\{T_{i,j} = g^{t_{i,j}}\}_{1 \leqslant j \leqslant n_i}\}_{1 \leqslant i \leqslant n}.$$

The public key PK is published as $PK = \langle e, g, g_1, g_2, g_3, Y, \{\{T_{i,j}\}_{1 \leqslant j \leqslant n_i}\}_{1 \leqslant i \leqslant n}\rangle$, the master key MK is $MK = \langle y, \{\{t_{i,j}\}_{1 \leqslant j \leqslant n_i}\}_{1 \leqslant i \leqslant n}\rangle$.

KeyGen(MK, L). Let $L = [L_1, L_2, \cdots, L_n]$ be the attribute list for the user who obtains the corresponding secret key. TA chooses $r_i \in_R \mathbb{Z}_p$ for $1 \leqslant i \leqslant n$, set $r = \sum_{i=1}^{n} r_i$ and computes $D_0 = g_2^{y-r}$. Then TA chooses $r' \in_R \mathbb{Z}_p$, sets $D'_0 = g^{r'}$ and computes D_i for $1 \leqslant i \leqslant n$ as $D_i = g_2^{r_i} T_{i,k_i}^{r'}$, (if $L_i = v_{i,k_i}$), TA outputs the secret key $SK_L = \langle L, D_0, D'_0, \{D_i\}_{1 \leqslant i \leqslant n}\rangle$.

Encrypt(PK, M, W). To encrypt a message $M \in \mathbb{G}_1$ under the access policy W, an encryptor chooses $s \in_R \mathbb{Z}_p$, computes $\tilde{C} = M \cdot Y^s$ and $C_0 = g^s, C'_0 = g_3^s$. Then the encryptor computes C_i for $1 \leqslant i \leqslant n$ as follows:

$$C_i = T_{i,k_i}^s \text{ (if } W_i = v_{i,k_i})$$

The encryptor computes $C' = \prod_{i=1}^{n} C_i$ and outputs the ciphertext

$$CT_W = \langle W, \tilde{C}, C_0, C'_0, C'\rangle.$$

RKGen(SK_L, W). Let SK_L denote a valid secret key, W an access policy. To generate a re-encryption key for W, chooses $d \in_R \mathbb{Z}_p$ and computes $g^d, D'_i = D_i g_3^d$. Leaving D_0 and D'_0 unchanged, computes $\mathbb{C}$ which is the ciphertext of $E(g^d)$ under the access policy W, i.e., $\mathbb{C} =$ **Encrypt**$(PK, E(g^d), W)$. Then the re-encryption key for W is $RK_{L \to W} = \langle L, W, D_0, D'_0, \{D'_i\}_{1 \leqslant i \leqslant n}, \mathbb{C}\rangle$.

Reencrypt$(RK_{L \to W'}, CT_W)$. Let $RK_{L \to W'}$ be a valid re-encryption key for access policy W', CT_W a well-formed ciphertext $\langle W, \tilde{C}, C_0, C'_0, C'\rangle$, checks W to know whether $L \models W$. If $L \not\models W$, returns $\perp$; otherwise, for $1 \leqslant i \leqslant n$, computes $D' = \prod_{i=0}^{n} D_i$ and $\bar{C} = \frac{e(C', D'_0)}{e(C_0, D')} = e(g, g_2)^{ys} e(g, g_3)^{nsd}$, outputs a re-encrypted ciphertext $CT' = \langle W', \tilde{C}, C'_0, \bar{C}, \mathbb{C}\rangle$.

Decrypt(CT_W, SK_L). A decryptor checks W to know whether $L \models W$. If $L \models W$, she can proceed. Then the decryptor decrypts the CT by using her SK_L as follows:

- If CT is an original well-formed ciphertext, then
  1. Compute $D' = \prod_{i=0}^{n} D_i$;
  2. $M = \tilde{C} \cdot e(C', D'_0)/e(C_0, D')$.
- Else if CT is a re-encrypted well-formed ciphertext, then
  1. Decrypts $E(g^d)$ from $\mathbb{C}$ using the secret key SK_L and decodes it to g^d,
  2. $M = \tilde{C} \cdot e(C'_0, g^d)^n / \bar{C}$.
- Else if CT is a multi-time re-encrypted well-formed ciphertext, decryption is similar with the above phases.

The security proofs are similar to previous scheme so we omit the detailed proof due to space consideration. Note that the new scheme has constant ciphertext size in original ciphertext and constant number of pairing computations in original decryption process.

4.2 Re-encryption Control

Note that if the encryptor does not provide g_3^s in ciphertext, the original decryption is not affected but the decryption of re-encrypted ciphertext cannot go on. That's because g_3^s is only used in decrypting re-encrypted ciphertext. So she can control whether the ciphertext can be re-encrypted. In the same way, the proxy can also decide whether the re-encrypted ciphertext can be re-encrypted.

If we want to remove this property, we should integrate g_3 into the secret key. For example, we can choose $r' \in_R \mathbb{Z}_p$, compute $D_0 = g_2^{y-r} g_3^{r'}$ and add $D'_0 = g^{r'}$ in the secret key. Then the decryption of original ciphertext should be modified as $M = \tilde{C} \cdot e(C'_0, D'_0) \prod_{i=1}^{n} e(C_i, D''_i) / (e(C_0, D_0) \prod_{i=1}^{n} e(C_0, D'_i))$. The modifications in other algorithms are similar and omitted here.

4.3 Extra Access Control

Our scheme allows the proxy to add extra access policy when re-encrypting ciphertext. For example, supposing the proxy can re-encrypt ciphertext under policy from W to W', he can add an extra access policy W'' to the re-encrypted ciphertext such that only user whose attribute list L simultaneously satifies W' and W'' can decrypt the re-encrypted ciphertext. He can achieve this as follows:

1. For a re-encryption key pair $D'_i = (D_{i,1} g_3^d,\ D_{i,2})$ or $F'_i = (F_{i,1} g_3^d,\ F_{i,2})$, choose a new $d' \in_R \mathbb{Z}_p$, compute new re-encryption key pair $D'_i = (D_{i,1} g_3^d g_3^{d'},\ D_{i,2})$ and $F'_i = (F_{i,1} g_3^d g_3^{d'},\ F_{i,2})$;
2. Use the new re-encryption key to re-encrypt ciphertext;
3. Add **Encrypt**$(PK, E(g^{d'}), W'')$ to the re-encrypted ciphertext.

4.4 Comparison

The properties of previous attribute-based proxy re-encryption schemes and our proposed schemes are compared in Table 1 and Table 2.

Table 1. Comparison: Some Properties

Scheme	Policy	Assumption	Security	Direction
GZWX [15]	Key	DBDH	Selective	Bidirectional
LCLS [19]	Ciphertext	ADBDH	Selective	Unidirectional
This Work 1	Ciphertext	DBDH	Selective	Unidirectional
This Work 2	Ciphertext	DBDH	Selective	Unidirectional

Table 2. Comparison: Expressiveness of policy

Scheme	Expressiveness
GZWX [15]	Tree-based Structure
LCLS [19]	AND-gates on positive and negative attributes with wildcards
This Work 1	AND-gates on multi-valued and negative attributes with wildcards
This Work 2	AND-gates on multi-valued attributes

5 Conclusions and Future Work

In this paper, we propose a novel ciphertext policy attribute-based proxy re-encryption scheme which is based on AND-gates policy supporting multi-value attributes, negative attributes and wildcards. Moreover, our scheme has two novel properties: 1) the encryptor can decide whether the ciphertext can be re-encrypted; 2) the proxy can add extra access policy during re-encryption process. And the proposed scheme can be modified to have constant ciphertext size in original encryption. We hope more rich access policies such as hidden policies, tree policies or access structures can be used in attribute-based proxy re-encryption schemes.

References

1. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. In: Proceedings of the Network and Distributed System Security Symposium, NDSS 2005. The Internet Society (2005)
2. Attrapadung, N., Imai, H.: Dual-policy attribute based encryption. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 168–185. Springer, Heidelberg (2009)
3. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, SP 2007, pp. 321–334 (2007)
4. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
5. Bobba, R., Khurana, H., Prabhakaran, M.: Attribute-sets: A practically motivated enhancement to attribute-based encryption. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 587–604. Springer, Heidelberg (2009)
6. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
7. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
8. Chase, M., Chow, S.S.: Improving privacy and security in multi-authority attribute-based encryption. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS 2009, pp. 121–130. ACM, New York (2009)
9. Cheung, L., Newport, C.: Provably secure ciphertext policy abe. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS 2009, pp. 456–465. ACM, New York (2007)
10. Chu, C.K., Tzeng, W.G.: Identity-based proxy re-encryption without random oracles. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 189–202. Springer, Heidelberg (2007)
11. Emura, K., Miyaji, A., Nomura, A., Omote, K., Soshi, M.: A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In: Bao, F., Li, H., Wang, G. (eds.) ISPEC 2009. LNCS, vol. 5451, pp. 13–23. Springer, Heidelberg (2009)
12. Goyal, V., Jain, A., Pandey, O., Sahai, A.: Bounded ciphertext policy attribute based encryption. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 579–591. Springer, Heidelberg (2008)

13. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, pp. 89–98. ACM, New York (2006)
14. Green, M., Ateniese, G.: Identity-based proxy re-encryption. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 288–306. Springer, Heidelberg (2007)
15. Guo, S., Zeng, Y., Wei, J., Xu, Q.: Attribute-based re-encryption scheme in the standard model. Wuhan University Journal of Natural Sciences 13(5), 621–625 (2008)
16. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
17. Lewko, A., Waters, B.: New techniques for dual system encryption and fully secure hibe with short ciphertexts. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 455–479. Springer, Heidelberg (2010)
18. Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
19. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute based proxy re-encryption with delegating capabilities. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security, ASIACCS 2009, pp. 276–286. ACM, New York (2009)
20. Mambo, M., Okamoto, E.: Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 80(1), 54–63 (1997)
21. Matsuo, T.: Proxy re-encryption systems for identity-based encryption. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 247–267. Springer, Heidelberg (2007)
22. Nishide, T., Yoneyama, K., Ohta, K.: Attribute-based encryption with partially hidden encryptor-specified access structures. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 111–129. Springer, Heidelberg (2008)
23. Okamoto, T., Takashima, K.: Hierarchical predicate encryption for inner-products. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 214–231. Springer, Heidelberg (2009)
24. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
25. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Cryptology ePrint Archive, Report 2008/290 (2008), `http://eprint.iacr.org/`
26. Waters, B.: Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)