Efficient Attribute-Based Proxy Re-Encryption with Constant Size Ciphertexts

Arinjita Paul(✉), S. Sharmila Deva Selvi, and C. Pandu Rangan

Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India
{arinjita,sharmila,prangan}@cse.iitm.ac.in

Abstract. Attribute-based proxy re-encryption (ABPRE) allows a semi-trusted proxy to transform an encryption under an access-policy into an encryption under a new access policy, without revealing any information about the underlying message. Such a primitive facilitates fine-grained secure sharing of encrypted data in the cloud. In its key-policy flavor, the re-encryption key is associated with an access structure that specifies which type of ciphertexts can be re-encrypted. Only two attempts have been made towards realising key-policy ABPRE (KP-ABPRE), one satisfying *replayable* chosen ciphertext security (RCCA security) and the other claiming to be chosen ciphertext secure (CCA secure). We show that both the systems are vulnerable to RCCA and CCA attacks respectively. We further propose a selective CCA secure KP-ABPRE scheme in this work. Since we demonstrate attacks on the only two existing schemes in the literature, our scheme becomes the first KP-ABPRE scheme satisfying selective CCA security. Moreover, our scheme has an additional attractive property, namely collusion resistance. A proxy re-encryption scheme typically consists of three parties: a delegator who delegates his decryption rights, a proxy who performs re-encryption, and a delegatee to whom the decryption power is delegated to. When a delegator wishes to share his data with a delegatee satisfying an access-policy, the proxy can collude with the malicious delegatee to attempt to obtain the private keys of the delegator during delegation period. If the private keys are exposed, security of the delegator's data is completely compromised. Hence, achieving *collusion resistance* is indispensable to real-world applications. In this paper, we show that our construction satisfies collusion resistance. Our scheme is proven collusion resistant and selective CCA secure in the random oracle model, based on Bilinear Diffie-Hellman exponent assumption.

Keywords: Proxy re-encryption · Key-policy · Attribute-based encryption · Random oracle · Bilinear map

K. Bhargavan et al. (Eds.): INDOCRYPT 2020, LNCS 12578, pp. 644–665, 2020.
https://doi.org/10.1007/978-3-030-65277-7_29

1 Introduction

In traditional proxy re-encryption (PRE) systems [1,4], the communication model is one-to-one, in the sense that a message can be re-encrypted only towards a particular public key. In practice, however many scenarios require the re-encryption functionality without exact knowledge of the set of intended recipients. One such major application is data sharing in untrusted cloud storage. In the cloud, a data owner may wish to share his encrypted data with users satisfying a specified access policy. In order to enable such expressive and fine-grained data sharing, Liang *et al.* [15] introduced the notion of attribute-based proxy re-encryption (ABPRE). ABPRE is an extension of the traditional PRE primitive to its attribute-based counterpart. PRE was introduced by Blaze, Bleumer and Strauss [4] to provide delegation of decryption rights. ABPRE designates a semi-trusted proxy to transform ciphertexts of delegators satisfying an access policy into ciphertexts of delegatees satisfying a new access policy. The proxy performs the conversion as a service upon receiving a special key construct called *re-encryption key* from the delegator, and while transforming the ciphertext, the proxy should not learn anything about the underlying message.

ABPRE integrates the notion of PRE with attribute-based encryption (ABE) to effectively enhance the flexibility of delegation of decryption capability. ABE is a generalization of Identity-Based Encryption (IBE), introduced in [22] wherein a user's identity is generalized to a set of descriptive attributes instead of a single string. ABE has two variants: *Key-Policy ABE* (KP-ABE) and *Ciphertext-Policy ABE* (CP-ABE). In KP-ABE systems, the private key captures an access structure that specifies which type of ciphertexts the key can decrypt, while ciphertexts are labeled with attribute sets. Its dual, CP-ABE associates a user's private key with attributes, while ciphertexts are embedded with access policy information. In light of the above, ABPRE can be classified into key-policy ABPRE (KP-ABPRE) and ciphertext-policy ABPRE (CP-ABPRE). Based on the direction of delegation, ABPRE systems are classified into unidirectional and bidirectional schemes. Additionally, based on the number of re-encryptions permitted, ABPRE systems are classified into single-hop and multi-hop schemes. In this work, we focus on unidirectional single-hop key-policy ABPRE schemes.

A cloud storage system may typically contain two kinds of encrypted data. The first kind termed *first-level ciphertext*, is the information that a user A encrypts under his attribute set W_1 and is likely to share with users identified by an attribute set W_2. Such an information is subject to re-encryption by the cloud, which performs the conversion upon getting the re-encryption key from user A. The second kind termed as *second-level ciphertext* is the re-encrypted data, converted by the cloud towards an access policy fulfilled by attribute set W_1 of user A, delegated by another user C. Such an encrypted file can not be further re-encrypted in a single hop scenario. The only way to illegally decrypt such ciphertexts is when a malicious user B with attributes W_2 and a cloud possessing a re-encryption key collude to obtain the private key corresponding to an access structure satisfied by W_1. Again, the re-encryption rights are enabled for a bounded, fixed period and malicious parties may want to decrypt ciphertexts of A even beyond that period. *Collusion attack* [1] refers to such an act where a colluding delegatee and cloud extract the private key of the delegator, causing harm to the delegator in every possible manner, such as unauthorised access to his sensitive data, identity theft and illegal delegation of decryption power. Thus, achieving collusion resistance is one of the major important problems in KP-ABPRE schemes. Collusion-resistant KP-ABPRE has many real-world applications such as blockchain based distributed cloud data storage and sharing, online medical service systems, online payment systems among others [14].

Achieving this powerful functionality of fine-grained access control comes at a cost. In a typical implementation of any previous construction of KP-ABPRE in the literature, the ciphertext size grows linearly with the size of the attribute set embedded by the sender. Also, the re-encryption and decryption time is proportional to the number of attributes involved of the receiver. Reducing ciphertext size and computation cost is highly beneficial in scenarios with low-bandwidth requirements and limited computing power. In this paper, we study KP-ABPRE in light of achieving collusion resistance and constant size ciphertexts.

Related Work and Contribution

In this work, we address the problem of designing collusion-resistant non-interactive attribute based PRE in the key-policy setting supporting rich access policies such as monotonic access structures, proposed in [13]. To integrate PRE to ABE setting, Liang *et al.* [15] first defined the notion of attribute-based PRE and proposed a CP-ABPRE scheme in the standard model, proven CPA-secure under Augment Decisional Bilinear Diffie-Hellman assumption. Chung *et al.* [6] gives a detailed study of the existing attribute based PRE schemes in their work. Li *et al.* [13] proposed the first KP-ABPRE scheme wherein matrix access structure is used to provide key-policy. Their construction is unidirectional, multi-use, collusion-resistant and is proven CPA-secure based on the Bilinear Decisional Diffie Hellman (DBDH) assumption. Note that, their construction relies on a trusted authority for the generation of the re-encryption keys, making the scheme highly infeasible. Since the same trusted authority is responsible for the generation of private keys, achieving delegation with the involvement of the trusted authority is trivial but impractical. Their work is extended in [12] to achieve an RCCA-secure KP-ABPRE scheme with matrix access structure to realize key-policy. Their design is unidirectional, multi-use and is claimed to be adaptively RCCA-secure based on DBDH assumption, with the same drawback of entrusting the trusted authority with the generation of re-encryption keys. Note that RCCA is a weaker variant of CCA tolerating a "harmless mauling" of the challenge ciphertext. In 2015, Ge *et al.* [11] designed a unidirectional single-hop CCA-secure KP-ABPRE scheme supporting monotonic access structures, without random oracles under the 3-weak decisional bilinear Diffie–Hellman inversion(3-wDBDHI) assumption. However, their construction does not adhere to the standard definition of KP-ABPRE. In essence, their scheme is a variation of conditional proxy re-encryption [23]. In their design, a first-level ciphertext

C is labelled with a set of descriptive conditions W, encrypted towards an individual public key pk_i, decryptable only using its corresponding private key sk_i. Here, every re-encryption key is generated using an access structure tree $\mathcal{T}$ associated with conditional keywords. That is, C can only be re-encrypted towards another public key pk_j specified in the re-encryption key $RK_{i,\mathcal{T},j}$ only if W (used to label C) satisfies the access tree $\mathcal{T}$ of the re-encryption key. Their system enables one-to-one communication subject to conditions specified via access trees in re-encryption key, rather than a many-to-many transformation enabled by KP-ABPRE. Recently, Ge *et al.* [10] proposed an adaptive CCA-secure collusion resistant KP-ABPRE scheme that supports any monotonic access structures on users' keys. Their scheme is claimed to be secure in the standard model based on subgroup decision problem for 3 primes and composite order bilinear pairings.

Our contribution in this work is twofold. Firstly, we demonstrate attacks on the security of two existing KP-ABPRE schemes in the literature. We show that the KP-ABPRE construction of Ge *et al.* [10] is vulnerable to CCA attack. We also demonstrate an RCCA attack on the KP-ABPRE scheme due to Li *et al.* [12], which claims to be RCCA secure. Consequently, only one result due to Li *et al.* [13] achieves KP-ABPRE, which is CPA secure in the random oracle model. In [7], Cohen *et al* remarks on the inadequacy of CPA security in PRE. Besides, the difficulty in achieving a CCA-secure KP-ABPRE scheme has been discussed in [12]. Our second contribution lies in designing the first construction of a selective CCA secure KP-ABPRE scheme in the selective model of security. Our scheme is proven secure under Decisional Bilinear Diffie Hellman Exponent assumption in the random oracle model. All the previous attempts to construct KP-ABPRE schemes admitting expressive re-encryption and decryption policies produce ciphertexts whose size grows linearly with the number of attributes embedded in the ciphertext for both levels of encryption. This paper proposes the first KP-ABPRE result allowing monotonic access structure with constant size ciphertext, based on the KP-ABE framework of Rao *et al.* [21] and BLS signature [5]. Also, the scheme enjoys the feature of constant number of exponentiations during encryption, and constant number of bilinear pairings during encryption, re-encryption and decryption. This is especially useful in applications that have low bandwidth requirements and limited computing power.

2 Preliminaries

2.1 Bilinear Maps and Hardness Assumptions

Definition 1 ***(Bilinear Maps).*** *Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be two finite cyclic groups of prime order p. A bilinear map is an efficient mapping $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ which is both: (bilinear) for all $a,\ b \in \mathbb{Z}_p^*$ and $g,\ h \in \mathbb{G}_0$, $\hat{e}(g^a, h^b) = \hat{e}(g,h)^{ab}$; and (non-degenerate) if $\mathbb{G}_0 = \langle g \rangle$, then $\mathbb{G}_1 = \langle \hat{e}(g,g) \rangle$.*

Definition 2 ***(n-Decisional Bilinear Diffie-Hellman Exponentiation assumption*** [21]***).*** *The n-decisional bilinear diffie-hellman exponentiation (n-DBDHE) assumption is, given the elements $\{g, g^b, g^a, g^{a^2}, \cdots, g^{a^n}, g^{a^{n+2}}, \cdots, g^{a^{2n}}\} \in \mathbb{G}_0$ and $Z \in \mathbb{G}_1$, there exists no PPT adversary which can decide whether*

$Z = \hat{e}(g,g)^{b(a^{n+1})}$ or a random element from $\mathbb{G}_1$ with a non-negligible advantage, where g is a generator of $\mathbb{G}_0$ and $a, b \in_R \mathbb{Z}_p^*$.

2.2 Access Structure and Linear Secret Sharing Schemes

Definition 3 ***(Access Structure*** [2]***).*** *Let $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\mathcal{P}}$ is monotone if $\forall B, C$, if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\mathcal{P}$, i.e., $\mathbb{A} \subseteq 2^{\mathcal{P}} \backslash \{\phi\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.*

In the context of ABE, the role of parties is taken by attributes. Thus, an access structure $\mathbb{A}$ contains all the authorized set of attributes. In this paper, we work with monotone access structures. Next, we define Linear Secret-Sharing Scheme (LSSS)-realizable access structure, used to specify access control policies over user attributes.

Definition 4 ***(Linear Secret Sharing Scheme).*** [3] *Let $\mathcal{P}$ be a set of parties. Let M be a matrix of size $l \times k$, called the share generating matrix. $\rho : [l] \rightarrow \mathcal{P}$ is a row-labeling function that maps rows in M to attributes in $\mathbb{A}$. A secret sharing scheme Π for access structure $\mathbb{A}$ is called linear in $\mathbb{Z}_p$ represented as (M, ρ) if it consists of the following polynomial time algorithms:*

- *Share(M, ρ, s): To generate shares of secret $s \in \mathbb{Z}_p$, it chooses $z_2, z_3, \cdots, z_k \xleftarrow{R} \mathbb{Z}_p$ and sets $v = (s, z_2, z_3, \cdots, z_k)^T$. It outputs $(M \cdot v)$ as the vector of l shares. Share $\lambda_i = (M_i \cdot v)$ belongs to attribute $\rho(i)$, where M_i^T is the i^{th} row of M.*
- *Reconstruct(M, ρ, W): This algorithm accepts as input $(M,\ \rho)$ and a set of attributes $W \in \mathbb{A}$. Let $I = \{i | \rho(i) \in W\}$. It outputs a set $\{\omega_i : i \in I\}$ of secret reconstruction constants such that $\sum_{i \in I} \omega_i \cdot \lambda_i = s$, if $\{\lambda_{\rho(i)} : i \in I\}$ is a valid set of secret shares of the secret s according to Π.*

3 Definition of KP-ABPRE

Definition 5 ***(Key-Policy Attribute-Based Proxy Re-Encryption (KP-ABPRE)).*** *A single-hop unidirectional KP-ABPRE scheme consists of the following seven algorithms:*

- ***Setup****$(1^\kappa, U)$: A PPT algorithm run by a certification authority CA that takes the unary encoding of the security parameter κ and an attribute universe description U as inputs. It outputs the public parameters as params and a master secret key MSK.*

- ***KeyGen***$(MSK, (M, \rho), params)$*: A PPT algorithm run by CA that takes as input the master secret key* MSK*, an access structure* (M, ρ) *for attributes over* U *and the public parameters* $params$*. It outputs a private key* $SK_{(M,\rho)}$ *corresponding to the access structure* (M, ρ)*.*
- ***Encrypt***$(m, W, params)$*: A PPT algorithm that takes as inputs a message* $m \in \mathcal{M}$*, an attribute set* W *and* $params$*. It outputs a ciphertext* C*, termed as first-level ciphertext, which can be further re-encrypted.*
- ***Decrypt***$(C, SK_{(M,\rho)}, params)$*: A deterministic algorithm that takes as input a first-level ciphertext* C *encrypted under attribute set* W*, a private key* $SK_{(M,\rho)}$*, and* $params$*. If* $W \models (M, \rho)$*, it outputs the message* $m \in \mathcal{M}$*, else output an error symbol* $\perp$ *indicating* C *is invalid.*
- ***ReKeyGen***$(SK_{(M,\rho)}, (M, \rho), (M', \rho'), params)$*: A PPT algorithm run by the delegator that takes as input a private key* $SK_{(M,\rho)}$ *corresponding to an access structure* (M, ρ)*, an access structure* (M', ρ') *and* $params$*. It outputs a re-encryption key* $RK_{(M,\rho) \to (M',\rho')}$*, that can perform re-encryption of ciphertexts under attribute set* $W \models (M, \rho)$ *towards attribute set* $W' \models (M', \rho')$*.*
- ***ReEncrypt***$(C, RK_{(M,\rho) \to (M',\rho')}, params)$*: A PPT algorithm run by proxy that takes as input a first-level ciphertext* C *encrypted under attribute set* W*, a re-encryption key* $RK_{(M,\rho) \to (M',\rho')}$ *and* $params$*. It outputs a re-encrypted ciphertext* D *if* $W \models (M, \rho)$ *or error symbol* $\perp$ *indicating* C *is invalid. The ciphertext* D *cannot be re-encrypted, also termed as second-level ciphertext.*
- ***ReDecrypt***$(D, SK_{(M',\rho')}, params)$*: A deterministic algorithm that takes as input a second-level ciphertext* D *encrypted under an attribute set* W'*, a private key* $SK_{(M',\rho')}$ *and* $params$*. If* $W' \models (M', \rho')$*, it outputs the message* $m \in \mathcal{M}$ *or an error symbol* $\perp$ *indicating* D *is invalid.*

The consistency of a KP-ABPRE scheme for any given public parameters $params$, private keys $SK_{(M,\rho)} \leftarrow KeyGen(MSK, (M, \rho), params)$, $SK_{(M',\rho')} \leftarrow KeyGen(MSK, (M', \rho'), params)$ and re-encryption keys $RK_{(M,\rho) \to (M',\rho')} \leftarrow ReKeyGen(SK_{(M,\rho)}, (M, \rho), (M', \rho'), params)$ and $\forall m \in \mathcal{M}$, the following equations hold:

1. Consistency between encryption and decryption:
   $Decrypt(C, SK_{(M,\rho)}, params) = m$, where $C \leftarrow Encrypt(m, W, params)$.
2. Consistency between encryption, proxy re-encryption and decryption:
   $ReDecrypt((ReEncrypt(C, RK_{(M,\rho) \to (M',\rho')}, params)), SK_{(M',\rho')}, params) = m$, where $C = Encrypt(m, W, params)$ and $W \models (M, \rho)$.

4 Security Model

Our game-based definitions of selective security of a single-hop unidirectional KP-ABPRE against CCA attack are adaptations of the definitions of CCA security for KP-ABPRE systems in [11]. A KP-ABPRE scheme is *IND-PRE-CCA* secure if no PPT adversary has a non-negligible advantage in the below game between the challenger $\mathcal{C}$ and adversary $\mathcal{A}$. In this model, the adversary $\mathcal{A}$ needs to fix the target access structure (M^*, ρ^*) beforehand in the Initialization phase.

Game Template of IND-PRE-CCA security:

- **Initialization:** $\mathcal{A}$ outputs a target access structure (M^*, ρ^*) on which it wishes to be challenged. $\mathcal{C}$ runs $Setup(1^\kappa, U)$ and sends the public parameter $params$ to $\mathcal{A}$.
- **Phase 1:** $\mathcal{A}$ issues queries to the following oracles simulated by $\mathcal{C}$:
  - Private Key Extraction($\mathcal{O}_{SK}(M, \rho)$): On input of an access structure (M, ρ), compute its corresponding private key $SK_{(M,\rho)}$ and return to $\mathcal{A}$.
  - Re-encryption Key Generation($\mathcal{O}_{RK}((M, \rho), (M', \rho'))$): Given two access structures (M, ρ) and (M', ρ')) as input, compute the re-encryption key $RK_{(M,\rho)\rightarrow(M',\rho')}$ and return it to $\mathcal{A}$.
  - Re-Encryption($\mathcal{O}_{RE}(C, (M, \rho), (M', \rho'))$): On input of a first-level ciphertext C, access structures (M, ρ) and (M', ρ'), compute re-encryption key $RK_{(M,\rho)\rightarrow(M',\rho')}$ and the second level ciphertext $D \leftarrow$ ReEncrypt($C, RK_{(M,\rho)\rightarrow(M',\rho')}, params$). Return D to $\mathcal{A}$.
  - Decryption($\mathcal{O}_{Dec}(C, (M, \rho))$): Given a first level ciphertext C and an access structure (M, ρ) as input, decrypt the ciphertext to obtain $m \in \mathcal{M}$. Return m or $\perp$ if the ciphertext is invalid.
  - Re-Decryption($\mathcal{O}_{RD}(D, (M', \rho'))$): Given a second level ciphertext D and an access structure (M', ρ') as input, decrypt the ciphertext to obtain $m \in \mathcal{M}$. Return m or $\perp$ if the ciphertext is invalid.
- **Challenge:** $\mathcal{A}$ decides whether it wants to be challenged with a first level or a second level ciphertext. It outputs two equal length messages m_0 and m_1 in $\mathcal{M}$ to $\mathcal{C}$. On receiving $\{m_0, m_1\}$, $\mathcal{C}$ picks $\psi \in \{0, 1\}$ at random, an attribute set $W^* \models (M^*, \rho^*)$ and generates a challenge ciphertext and returns to $\mathcal{A}$.
- **Phase 2:** $\mathcal{A}$ issues queries to the oracles similar to Phase 1, subject to constraints as discussed later.
- **Guess:** $\mathcal{A}$ outputs its guess $\psi' \in \{0, 1\}$.

The actual construction of challenge ciphertext and constraints on queries of $\mathcal{A}$ are defined based on the ciphertext type that $\mathcal{A}$ opts for. Due to the existence of two levels of ciphertexts, namely *first level* and *second level*, the security game and different adversarial constraints at both levels are shown next.

4.1 First Level Ciphertext Security:

For the first-level ciphertext security, $\mathcal{C}$ interacts with $\mathcal{A}$ as per the game template shown above (Game 1), with the following adversarial constraints, where the challenge ciphertext is C^* =Encrypt($m_\psi, W, params$) and $W \models (M^*, \rho^*)$:

- $\mathcal{O}_{SK}(M^*, \rho^*)$ should not be queried by $\mathcal{A}$.
- $\mathcal{O}_{RK}((M^*, \rho^*), (M', \rho'))$ must not be queried if $\mathcal{O}_{SK}(M', \rho')$ has already been queried.
- $\mathcal{O}_{SK}(M', \rho')$ must not be queried if $\mathcal{O}_{RE}(C^*, (M^*, \rho^*), (M', \rho'))$ has already been queried.
- $\mathcal{O}_{SK}(M', \rho')$ must not be queried if $\mathcal{O}_{RK}((M^*, \rho^*), (M', \rho'))$ has already been queried.

CCA-Security Game 1 $Exp_{A,1}^{IND-PRE-CCA}(\kappa)$

$(M^*, \rho^*) \leftarrow \mathcal{A}$, params $\leftarrow$ Setup(κ);
$(m_0, m_1, St) \leftarrow \mathcal{A}^{\mathcal{O}_{SK}, \mathcal{O}_{RK}, \mathcal{O}_{RE}, \mathcal{O}_{Dec}, \mathcal{O}_{RD}}(params)$;
$\psi \ \epsilon_R \ \{0,1\}, \ \mathbb{C}^* \leftarrow$Encrypt($m_\psi, W \models (M^*, \rho^*)$,params);
$\psi' \leftarrow \mathcal{A}^{\mathcal{O}_{RK}, \mathcal{O}_{RE}, \mathcal{O}_{Dec}, \mathcal{O}_{RD}}(\mathbb{C}^*, St)$; //query constraints are shown in Section 4.1
if $\psi = \psi'$ return 1, else return 0

Game 1: IND-PRE-CCA security game for first level ciphertext. Note that *St* is the state information maintained by $\mathcal{A}$.

CCA-Security Game 2 $Exp_{A,2}^{IND-PRE-CCA}(\kappa)$

$(M^*, \rho^*) \leftarrow \mathcal{A}$, params $\leftarrow$ Setup(κ);
$(m_0, m_1$,St$) \leftarrow \mathcal{A}^{\mathcal{O}_{SK}, \mathcal{O}_{RK}, \mathcal{O}_{Dec}, \mathcal{O}_{RD}}(params)$;
$\psi \ \epsilon_R \ \{0,1\}, \ \mathbb{C}^* \leftarrow$Encrypt($W \models (M^*, \rho^*), m_\psi$,params);
$\psi' \leftarrow \mathcal{A}^{\mathcal{O}_{RK}, \mathcal{O}_{Dec}, \mathcal{O}_{RD}}(\mathbb{C}^*, St)$; //query constraints are shown in Section 4.2
if $\psi = \psi'$ return 1, else return 0

Game 2: IND-PRE-CCA security game for second level ciphertext.

- $\mathcal{O}_{RE}(C^*, (M^*, \rho^*), (M', \rho'))$ must not be queried if $\mathcal{O}_{SK}((M', \rho'))$ has already been queried.
- $\mathcal{O}_{Dec}(C^*, (M^*, \rho^*))$ cannot be queried by $\mathcal{A}$.
- $\mathcal{O}_{RD}(D, (M', \rho'))$ cannot be queried for second level ciphertext D re-encrypted towards (M', ρ') where D is a *challenge derivative* (defined next) of C^*.

Definition 6 ***Challenge Derivative***[10]. *A challenge derivative of C^* in the CCA setting is inductively defined as below:*

- *Reflexivity: C^* is a challenge derivative of itself.*
- *Derivative by re-encryption: D is a challenge derivative of C^* if $D \leftarrow \mathcal{O}_{RE}(C^*, (M^*, \rho^*), (M', \rho'))$.*
- *Derivative by re-encryption key: D is a challenge derivative of C^* if $RK_{(M^*,\rho^*)\rightarrow(M',\rho')} \leftarrow \mathcal{O}_{RK}((M, \rho), (M', \rho'))$ and $D = ReEncrypt(C^*, RK_{(M,\rho)\rightarrow(M',\rho')}, params)$.*

Definition 7. *The advantage of any PPT adversary $\mathcal{A}$ denoted by $Adv_{\mathcal{A}}$ in winning the above IND-PRE-CCA game (Game 1) for first level ciphertext which we term as $Exp_{A,1}^{IND-PRE-CCA}(\kappa)$ is shown as*

$$Adv_{\mathcal{A},1}^{IND-PRE-CCA} := \left| Pr\Big[Exp_{A,1}^{IND-PRE-CCA}(\kappa)\Big] - \frac{1}{2} \right|$$

where the probability is over the coin tosses of challenger $\mathcal{C}$ and adversary $\mathcal{A}$.

The scheme is $IND\text{-}PRE\text{-}CCA$ secure for the first level ciphertext against any t-time adversary $\mathcal{A}$ that makes atmost q_{SK}, q_{RK}, q_{ReEnc}, q_{Dec} and q_{RD} queries to $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_{RE}$, $\mathcal{O}_{Dec}$ and $\mathcal{O}_{RD}$ oracles respectively, if the advantage of $\mathcal{A}$ is negligibly small: $Adv_{\mathcal{A},1}^{IND-PRE-CCA} \leq \epsilon$.

4.2 Second Level Ciphertext Security:

For the second-level ciphertext security game (Game 2), the adversarial constraints on $\mathcal{A}$ are given below, where the challenge ciphertext is $D^* = ReEncrypt(C, RK_{(M',\rho')\rightarrow(M^*,\rho^*)}, params)$ and C is a first level encryption of m_ψ under the delegator's attribute set W' satisfying access structure (M', ρ'). Note that for single-hop KP-ABPRE schemes, $\mathcal{A}$ is given access to all possible re-encryption keys. As a result, there is no need to provide $\mathcal{A}$ with the re-encryption oracle.

- $\mathcal{O}_{SK}((M^*, \rho^*))$ should not be queried by $\mathcal{A}$.
- $\mathcal{O}_{RD}(D^*, (M^*, \rho^*))$ must not be queried.

Definition 8. *The advantage of any PPT adversary $\mathcal{A}$ denoted by $Adv_{\mathcal{A}}$ in winning the above IND-PRE-CCA game (Game 2) for second level ciphertext which we term $Exp_{A,2}^{IND-PRE-CCA}(\kappa)$ is shown as*

$$Adv_{\mathcal{A},2}^{IND-PRE-CCA} := \left| Pr\left[Exp_{A,2}^{IND-PRE-CCA}(\kappa) \right] - \frac{1}{2} \right|$$

where the probability is over the coin tosses of challenger $\mathcal{C}$ and adversary $\mathcal{A}$.

The scheme is $IND\text{-}PRE\text{-}CCA$ secure for the second level ciphertext against any t-time adversary $\mathcal{A}$ that makes atmost q_{SK}, q_{RK}, q_{ReEnc}, q_{Dec} and q_{RD} queries to $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_{Dec}$ and $\mathcal{O}_{RD}$ oracles respectively, if the advantage of $\mathcal{A}$ is negligibly small: $Adv_{\mathcal{A},2}^{IND-PRE-CCA} \leq \epsilon$.

DSK-Security Game $Exp_A^{DSK}(\kappa)$

$(M^*, \rho^*) \leftarrow \mathcal{A}$, params $\leftarrow$ Setup(κ);

$(SK_{(M^*,\rho^*)}, St) \leftarrow \mathcal{A}^{\mathcal{O}_{SK}, \mathcal{O}_{RK}}(params)$; //query constraints shown in Section 4.3

if $SK_{(M^*,\rho^*)}$ is a valid private key of (M^*, ρ^*), return 1, else return 0

Game 3: DSK security game for KP-ABPRE schemes.

4.3 Collusion Resistance

Collusion-resistance, also termed as delegator secret key security (DSK security) prevents a colluding proxy and delegatee to recover private keys of the delegator in full. The game template for DSK security is shown below, adapted from [18], illustrated in Game 3.

- **Setup**: $\mathcal{A}$ outputs a challenge access structure (M^*, ρ^*). $\mathcal{C}$ generates the public parameters *params* using the *Setup* algorithm and returns it to $\mathcal{A}$.
- **Queries**: $\mathcal{A}$ issues queries to the Private Key Extraction($\mathcal{O}_{SK}(M, \rho)$) and Re-encryption Key Generation($\mathcal{O}_{RK}((M, \rho), (M', \rho'))$) oracle adaptively. It cannot query for the private key of the target access structure (M^*, ρ^*).
- **Output**: $\mathcal{A}$ returns $SK_{(M^*,\rho^*)}$ as the private key of the target access structure (M^*, ρ^*). $\mathcal{A}$ wins the game if $SK_{(M^*,\rho^*)}$ is a valid private key of (M^*, ρ^*).

Definition 9. *The advantage of any PPT adversary $\mathcal{A}$ denoted by $Adv_{\mathcal{A}}$ in winning the DSK-security game given above (Game 3) which we term $Exp_A^{DSK}(\kappa)$ is shown as*

$$Adv_{\mathcal{A}}^{DSK} := \Pr[\mathcal{A}\ wins]$$

where the probability is over the coin tosses of challenger $\mathcal{C}$ and adversary $\mathcal{A}$.

The scheme is DSK secure against any t-time adversary $\mathcal{A}$ that makes atmost q_{SK} and q_{RK} queries to $\mathcal{O}_{SK}$ and $\mathcal{O}_{RK}$ oracles respectively, if the advantage of $\mathcal{A}$ is negligibly small: $Adv_{\mathcal{A}}^{DSK} \leq \epsilon$.

5 Analysis of a CCA-secure KP-ABPRE Scheme

5.1 Review of the Scheme Due to Ge *et al.* [10]

The CCA-secure KP-ABPRE scheme due to Ge *et al* [10] consists of the following algorithms. It is based on composite order bilinear pairing.

- **Setup**$(1^\kappa, U)$: The setup algorithm chooses a bilinear group $\mathbb{G}_0$ of order $N = p_1p_2p_3$. Let $\mathbb{G}_{p_i}$ denote the subgroup of $\mathbb{G}_0$ of order p_i. It chooses $\alpha, \phi \in \mathbb{Z}_N$, $g, \hat{g_1}, \hat{g_2} \in \mathbb{G}_{p_1}$, $X_3 \in \mathbb{G}_{p_3}$. For each attribute i, it picks $s_i \in \mathbb{Z}_N$ and computes $T_i = g^{s_i}$. Let SYM be a CCA-secure one-time symmetric encryption scheme, and Sig $= (\mathcal{G}, \mathcal{S}, \mathcal{V})$ be a strongly unforgeable one-time signature scheme. $H_1 : \mathbb{G}_1 \rightarrow \mathbb{Z}_N^*$ and $H_2 : \mathbb{G}_1 \rightarrow \{0, 1\}^*$ are collision-resistant hash functions. The master secret key is $MSK = (\alpha, X_3)$. The public parameters are $params = (N, g, \hat{g_1}, \hat{g_2}, \hat{g_2}^{\phi}, \hat{e}(g, g)^\alpha, T_i, SYM, (\mathcal{G}, \mathcal{S}, \mathcal{V}), \mathcal{H}_1, \mathcal{H}_2)$.
- **KeyGen**$(MSK, (M, \rho), params)$: Given as input an access structure (M, ρ), the trusted authority picks a random vector μ such that $\mu \cdot \mathbf{1} = \alpha$. For each row M_x of matrix M, it chooses $r_x \in \mathbb{Z}_N$ and $W_x, V_x \in \mathbb{G}_{p_3}$ and computes private key as: $\forall M_x \in \{M_1, \cdots M_l\}$: $K_{x,1} = g^{M_x \cdot \mu} T_{\rho(x)}^{r_x} W_x$, $K_{x,2} = g^{r_x} V_x$.
- **Encrypt**$(m, W, params)$: To encrypt a message $m \in \mathbb{G}_1$ under an attribute set W, the sender encrypts as shown below:
  1. Set $C_1 = W$. Select a one-time signature pair $(svk, ssk) \leftarrow \mathcal{G}$.
  2. Pick $s \in \mathbb{Z}_N$ and compute $C_0 = m \cdot \hat{e}(g, g)^{\alpha s}, C_2 = g^s, \forall i \in W$: $C_i = T_i^s, C_3 = (\hat{g_1}^{svk} \hat{g_2}^{\phi})^s$. Run the sign algorithm $\sigma \leftarrow \mathcal{S}(ssk, (C_0, C_2, C_i, C_3))$.
  3. Return the original ciphertext $C = (svk, C_0, C_1, C_2, C_i, C_3, \sigma)$.

- **ReKeyGen**$(SK_{(M,\rho)}, (M,\rho), (M',\rho'), params)$: The delegator generates a re-encryption key from access structure (M,ρ) to (M',ρ'), as shown below:
  1. Choose $\theta \in \mathbb{Z}_p$ and $\delta \in \mathbb{G}_1$. For each row M_x of the matrix M, compute:
  $$rk_1 = K_{x,1}^{H_1(\delta)} \cdot T_{\rho(x)}^{\theta}, rk_2 = K_{x,2}^{H_1(\delta)}, rk_3 = g^{\theta}.$$
  2. Select an attribute set W' where $W' \models (M',\rho')$.
  3. Select a one-time signature pair $(svk', ssk') \leftarrow \mathcal{G}$.
  4. Pick $s' \in \mathbb{Z}_N$ and compute $rk_4 = \delta \cdot \hat{e}(g,g)^{\alpha s'}, rk_5 = g^{s'}, \forall i \in W' : rk_{6,i} = T_i^{s'}, rk_7 = (\hat{g_1}^{svk'} \hat{g_2}^{\phi})^{s'}$.
  5. Run the sign algorithm $\sigma' \leftarrow \mathcal{S}(ssk', (rk_4, rk_5, rk_{6i}, rk_7))$.
  6. Return $RK_{(M,\rho)\to(M',\rho')} = (rk_1, rk_2, rk_3, S', svk', rk_4, rk_5, rk_{6,i}, rk_7, \sigma')$.
- **ReEncrypt**$(C, RK_{(M,\rho)\to(M',\rho')}, params)$: On input of an original ciphertext C, the proxy re-encrypts C towards access structure (M',ρ') as below:
  1. Check if the following equations hold:
  $$\mathcal{V}(svk, \sigma, (C_0, C_2, C_i, C_3)) \stackrel{?}{=} 1, \quad \hat{e}(C_2, \hat{g_1}^{svk} \hat{g_2}^{\phi}) \stackrel{?}{=} \hat{e}(g, C_3).$$
  2. If the above check fails, return $\perp$. Else, if $W \models (M,\rho)$, compute reconstruction constants ω_x such that $\sum_{\rho(x)\in W} \omega_x M_x = 1$. Compute $Q = \prod \frac{\hat{e}(C_2, rk_1)^{\omega_x}}{(\hat{e}(C_{\rho_x}, rk_2)\hat{e}(C_{\rho(x)}, rk_3))^{\omega_x}}$.
  3. Pick a random $key \in \mathbb{G}_1$ and compute $\varPhi_1 = SYM.Enc(H_2(key, G))$, where $G = (C||(W', svk', rk_4, rk_5, rk_{6,i}, rk_7, \sigma')||Q)$.
  4. Select an attribute set W'' such that $W'' \models (M',\rho')$ and a one-time signature pair $(svk'', ssk'') \leftarrow \mathcal{G}$. Choose $s'' \in \mathbb{Z}_N$ and compute $C_0'' = key \cdot \hat{e}(g,g)^{\alpha s''}$, $C_2'' = g^{s''}$, $\forall i \in W'' : C_i'' = T_i^{s''}, C_3'' = (\hat{g}_1^{svk''} \hat{g}_2^{\phi})^{s''}$. Run the sign algorithm to generate $\sigma'' \leftarrow \mathcal{S}(ssk'', (C_0'', C_2'', C_i'', C_3''))$. Denote $\varPhi_2 = (W'', svk'', C_0'', C_2'', C_i'', C_3'', \sigma'')$.
  5. Return the re-encrypted ciphertext $D = (\varPhi_1, \varPhi_2)$.
- **Decrypt**$(C, SK_{(M,\rho)}, params)$: In order to decrypt a first-level ciphertext C, the decryption algorithm proceeds as below:
  1. Check the validity of the ciphertext using Eqs. (1) and (2).
  2. If check fails, output $\perp$ and aborts. Otherwise if $W \not\models (M,\rho)$, output $\perp$ and aborts. Else, compute reconstruction constants ω_x such that $\sum_{\rho(x)\in W} \omega_x M_x = 1$. Compute plaintext $m = C_0 \Big/ \prod_{\rho(x)\in S} \frac{\hat{e}(C_2, K_{x,1})^{\omega_x}}{\hat{e}(C_{\rho(x)}, K_{x,2})^{\omega_x}}$.
- **ReDecrypt**$(D, sk_{(M',\rho')}, params)$: To decrypt a second-level ciphertext $D = (\varPhi_1, \varPhi_2)$, the decryption algorithm proceeds as below:
  - Check if the following equations hold:
  $$\mathcal{V}(svk'', \sigma'', (C_0'', C_2'', C_i'', C_3'')) \stackrel{?}{=} 1, \quad \hat{e}(C_2'', \hat{g}_1^{svk''} \hat{g}_2^{\phi}) \stackrel{?}{=} \hat{e}(g, C_3'').$$

  If the checks fail, output $\perp$ and abort. Further, if $W'' \not\models M''$, output $\perp$ and abort. Compute the reconstruction constants ω'' such that $\sum_{\rho(x')\in W''} \omega_x'' M_x'' = 1$. Compute $key = C_0'' \Big/ \prod_{\rho(x')\in W''} \frac{\hat{e}(C_2'', K_{x',1})^{\omega_x''}}{\hat{e}(C''_{\rho x'}, K_{x',2})^{\omega_x''}}$.

• Run the decryption algorithm $G = SYM.Dec(H_2(key), \Phi_1)$.
• Check if the Eqs. (1) and (2) hold. If fails, return $\perp$ and abort. Otherwise, perform the following checks:

$$\mathcal{V}(svk', \sigma', (rk_4, rk_5, rk_{6,i}, rk_7)) \overset{?}{=} 1, \quad \hat{e}(rk_5, \hat{g}_1^{svk'} \hat{g}_2^{\phi}) \overset{?}{=} \hat{e}(g, rk_7).$$

If fails, return $\perp$ and abort. If $W' \not\models (M', \rho')$, return $\perp$ and abort.
• Compute the reconstruction constants ω'_x such that $\sum_{\rho(x') \in W'} \omega'_x M'_x = 1$. Next, compute $\delta = rk_4 \Big/ \prod_{\rho(x') \in W'} \frac{\hat{e}(rk_5, K_{x',1})^{\omega'_x}}{\hat{e}(rk_{6,\rho(x')}, K_{x',2})^{\omega'_x}}$.
• Compute $Q^{H_1(\delta)^{-1}} = \hat{e}(g,g)^{s\alpha}$ and output the plaintext $m = C_0 / \hat{e}(g,g)^{s\alpha}$.

5.2 Attack on the Scheme

In this section we present a CCA-attack on the scheme due to Ge *et al* [10]. Note that the following attack is launched in the second-level ciphertext CCA-security game. In Phase 1 of the security game, the challenger $\mathcal{C}$ provides the adversary $\mathcal{A}$ with all possible re-encryption keys in the system, as per the security definition in [10]. Let $D^* = (\Phi_1^*, \Phi_2^*)$ be the challenge ciphertext generated by $\mathcal{C}$, which is the re-encryption of message m_ψ (selected randomly by $\mathcal{C}$ during challenge phase) from a delegator's attribute set satisfying access structure (M', ρ') towards target access structure (M^*, ρ^*). The CCA attack is demonstrated below.

1. $\mathcal{A}$ parses re-encryption key $RK_{(M',\rho') \to (M^*,\rho^*)} = (rk_1^*, rk_2^*, rk_3^*, W^*, svk'^*, rk_4^*, rk_5^*, rk_{6,i}^*, rk_7^*, \sigma'^*)$.
2. $\mathcal{A}$ creates a first-level decryption query for a ciphertext generated as below:
   - Set $C_0 = rk_4^*$, $C_1 = W^*$, $C_2 = rk_5^*$ and $C_3 = rk_7^*$.
   - For all attributes $i \in W^*$, set $C_i = rk_{6,i}^*$. Set the signature $\sigma = \sigma'^*$.
   - The ciphertext $C_{A1} = (svk'^*, C_0, C_1, C_2, C_i, C_3, \sigma)$ is passed a parameter to the first level decryption oracle provided by the challenger.
3. $\mathcal{C}$ decrypts ciphertext C_{A1} using Decrypt algorithm to extract δ^* used in re-encryption key generation corresponding to challenge ciphertext D^*.
4. $\mathcal{A}$ parses challenge ciphertext component $\Phi_2^* = (W''^*, svk''^*, C_0''^*, C_2''^*, C_i''^*, C_3''^*, \sigma''^*)$.
5. Using this second-level challenge ciphertext, the adversary now creates another decryption query for a first level ciphertext generated as below:
   - Set $C_0 = C_0''^*$, $C_1 = W''^*$, $C_2 = C_2''^*$ and $C_3 = C_3''^*$.
   - For all attributes $i \in W''^*$, set $C_i = C_i''^*$. Set the signature $\sigma = \sigma''^*$.
   - The ciphertext $C_{A2} = (svk''^*, C_0, C_1, C_2, C_i, C_3, \sigma)$ is passed a parameter to the first level decryption oracle provided by the challenger.
6. On decryption of C_{A2}, $\mathcal{A}$ receives key^* used in generation of the challenge ciphertext D^*. Therefore, $\mathcal{A}$ can now recover G^* using the symmetric decryption algorithm $G^* = SYM.Dec(H_2(key^*), \Phi_1^*)$.
7. $\mathcal{A}$ parses $G^* = (C||(W^*, svk^*, rk_4^*, rk_5^*, rk_{6,i}^*, rk_7^*, \sigma^*)||Q)$ and then parses $C = (svk, C_0, C_1, C_2, C_i, C_3, \sigma)$.

8. Finally $\mathcal{A}$ computes $m_\psi = \frac{C_0}{Q^{H_1(\delta)^{-1}}}$ as per ReDecrypt() algorithm.

This completes the description of the attack. $\mathcal{A}$ can recover the original message m_ψ re-encrypted by $\mathcal{C}$ towards a target access structure (M^*, ρ^*) in the challenge ciphertext D^*, which successfully breaks the CCA security of the scheme.

6 Analysis of an RCCA-secure KP-ABPRE Scheme

6.1 Review of the Scheme Due to Li *et al.* [12]

The RCCA-secure KP-ABPRE scheme due to Li *et al.* [12] consists of the following algorithms.

- **Setup**$(1^\kappa, U)$: The setup algorithm takes as input the universe description U, where $U = \{0,1\}^*$ and security parameter κ. It chooses groups $\mathbb{G}_0, \mathbb{G}_1$ of prime order p. Let g by a generator of $\mathbb{G}_0$. It randomly picks values $\alpha \in \mathbb{Z}_p$ and $g_1, h \in \mathbb{G}_0$, and sets $MSK = \alpha$. k is a parameter determined by κ and $\{0,1\}^k$ is the message space $\mathcal{M}$. Three cryptographic hash functions are chosen as follows: $F : \{0,1\}^* \rightarrow \mathbb{G}_0$, $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^k$. The public parameters $params = \langle g, g_1, h, g^\alpha, F, H_1, H_2, \hat{e}(g,h)^\alpha \rangle$.
- **KeyGen**$(MSK, (M,\rho), params)$: On input of an access structure (M,ρ) where $l \times k$ is the size of access structure M, the trusted authority executes Share(M, ρ, α) to obtain l sets of shares $\lambda_{\rho(i)} = M_i \cdot v$, where it picks $y_2, \cdots y_n \in \mathbb{Z}_p$ and sets $v = (\alpha, y_2, \cdots y_n)$. It chooses $r_1, \cdots r_l \in \mathbb{Z}_p^*$ and computes the private key $SK_{(M,\rho)}$ as follows: $K_{11} = h^{\lambda_{\rho(1)}} \cdot F(\rho(1))^{r_1}, K_{21} = g^{r_1}, \cdots, K_{1l} = h^{\lambda_{\rho(l)}} \cdot F(\rho(l))^{r_l}, K_{2l} = g^{r_l}$. It outputs $SK_{(M,\rho)}$ along with the description of (M,ρ).
- **Encrypt**$(m, W, params)$: Given as input a message $m \in \mathcal{M}$ and an attribute set W, the algorithm randomly picks $R \in \mathbb{G}_1$ and computes $s_1 = H_1(R, m)$ and $r = H_2(R)$. It computes the first level encryption C as follows: $C_0 = R \cdot \hat{e}(g,h)^{\alpha s_1}$, $C_1 = g^{s_1}$, $C_2 = g_1^{s_1}$, $C_3 = r \oplus m, \{C_x = F(x)^{s_1}, \forall x \in W\}$. It outputs ciphertext $C = \langle C_0, C_1, C_2, C_3, \{C_x, \forall x \in W\}, W \rangle$.
- **ReKeyGen**$(SK_{(M,\rho)}, (M,\rho), (M',\rho'), params)$: To generate a re-encryption key from an access structure (M,ρ) to (M',ρ'), the trusted authority calls the **KeyGen** algorithm and chooses $d \in \mathbb{Z}_p^*$ at random and computes $g^{\alpha d}$, $\{g_1^{\lambda_{\rho(i)} d}\}$ where $\lambda_\rho(i)$ for $i = 1$ to l is the set of l shares corresponding to access structure (M,ρ). It picks an attribute $W' \models (M',\rho')$ and encrypts $g^{\alpha d}$ with the attributes of W' by computing $CT_1 = \textbf{Encrypt}(g^{\alpha d}, W', params)$. It picks random $r_1' \cdots r_l' \in \mathbb{Z}_p$ and recomputes key $RK_{(M,\rho)\rightarrow(M',\rho')}$ as follows: $rk_{11} = h^{\lambda_{\rho(1)}} \cdot F(\rho(1))^{r_1'} \cdot g_1^{\lambda_{\rho(1)} d}, rk_{21} = g^{r'_1}, \cdots, rk_{1l} = h^{\lambda_{\rho(l)}} \cdot F(\rho(l))^{r_l'} \cdot g_1^{\lambda_{\rho(l)} d}, rk_{2l} = g^{r_l'}$. It returns the re-encryption key as $RK_{(M,\rho)\rightarrow(M',\rho')} = (\{rk_{1i}, rk_{2i}$ for i=1 to $l\}, CT_1)$.
- **ReEncrypt**$(C, RK_{(M,\rho)\rightarrow(M',\rho')}, params)$: Given as input a first-level ciphertext C and a re-encryption key $RK_{(M,\rho)\rightarrow(M',\rho')}$, the re-encryption algorithm executes Reconstruct (M, ρ, W) to obtain a set $\{\omega_i : i \in I\}$ of secret reconstruction constants where $I = \{i \in [l] : att_{\rho(i)} \in W\}$. If $W \models (M,\rho)$, then

the relation $\sum_{i \in I} \omega_i \lambda_{\rho(i)} = \alpha$ implicitly holds. It computes the re-encrypted ciphertext component $D_4 = \frac{\hat{e}(C_1, \prod_{i \in I} rk_{1i}^{\omega_i})}{\prod_{i \in I} \hat{e}(rk_{2i}, C_{\rho(i)}^{\omega_i})} = \hat{e}(g,h)^{s_1 \alpha} \hat{e}(g, g_1)^{(s_1 \alpha d)}$. It sets $D_0 = C_0, D_1 = C_1, D_2 = C_2, D_3 = C_3, D_5 = CT_1$ and outputs second level ciphertext $D = \langle D_0, D_1, D_2, D_3, D_4, D_5 \rangle$.

- **Decrypt**(C or $D, SK_{(M', \rho')}, params$): If the input to this algorithm is a first level ciphertext C encrypted under (M, ρ), the decryption algorithm invokes **ReEncrypt**($C, RK_{(M,\rho) \to (M', \rho')}, params$) to obtain $D = \langle D_0, D_1, D_2, D_3, D_4, D_5 \rangle$. If the input is a second level ciphertext D and a private key $SK_{(M', \rho')}$, the algorithm first decrypts $D_5 = \langle C'_0, C'_1, C'_2, C'_3, \{C'_x\}, W' \rangle$ by checking if $W' \models (M', \rho')$ and computing the set of reconstruction constants $\{\omega'_i : i \in I\}$ where $I = \{i \in [l] : att_{\rho(i)'} \in W'\}$ using Reconstruct (M', ρ', W') such that $\sum_{\rho'(i) \in W'} \omega'_i M'_i = 1$ holds implicitly. It computes $CT_2 = \frac{\hat{e}(C'_1, \prod_{i \in I} K_{1i}^{\omega'_i})}{\prod_{i \in I} \hat{e}(K_{2i}, C'^{\omega'_i}_{\rho(i)})} = \hat{e}(g,h)^{s_1 \alpha}$. It extracts $g^{\alpha d} = D_0 / CT_2$. It computes $CT_3 = \hat{e}(g^{\alpha d}, D_2) = \hat{e}(g, g_1)^{s_1 \alpha d}$. Next, it computes $R = D_0 \cdot CT_3 / D_4$, $m = D_3 \oplus H_2(R)$ and $s = H_1(R, m)$. If $D_0 \overset{?}{=} R \cdot \hat{e}(g,h)^{\alpha s}$ and $D_4 \overset{?}{=} \hat{e}(g,h)^{s_1 \alpha} \cdot \hat{e}(g^{\alpha d}, g_1^{s_1)}$, it outputs m. otherwise return $\perp$.

6.2 Attack on the Scheme

In this section, we present an RCCA-attack on the scheme due to Li *et al* [12]. The following attack is launched in the first-level ciphertext RCCA-security game. Suppose that $C^* = \langle C^*_0, C^*_1, C^*_2, C^*_3, \{C^*_x\}, W^* \rangle$ is the challenge ciphertext generated by $\mathcal{C}$, which is the encryption of message m_ψ (selected randomly by $\mathcal{C}$ during challenge phase from messages $\{m_0, m_1\}$) towards a delegator's attribute set $W^* \models (M^*, \rho^*)$ where (M^*, ρ^*) is the target access structure. The RCCA attack launched by the adversary $\mathcal{A}$ is demonstrated below:

1. The adversary $\mathcal{A}$ picks $\beta \in \mathbb{Z}_p^*$ at random.
2. It computes C_0" $= C^*_0 \cdot \hat{e}(g,h)^{\alpha \beta} = R \cdot \hat{e}(g,h)^{\alpha(s^*_1 + \beta)}$.
3. It computes C_1" $= C^*_1 \cdot g^\beta = g^{s^*_1 + \beta}$ and C_2" $= C^*_2 \cdot g_1^\beta = g_1^{s^*_1 + \beta}$.
4. It picks C_3" $\in \{0,1\}^k$ at random.
5. For all $x \in W^*$, it computes $\{C_x$" $= C^*_x \cdot F(x)^\beta = F(x)^{s^*_1 + \beta}\}$.
6. It constructs a first level ciphertext C" $= \langle C_0$", C_1", C_2", C_3", $\{C_x$"$\}, W^* \rangle$.
7. It queries the re-encryption oracle $\mathcal{O}_{RE}(C$", $(M^*, \rho^*), (M', \rho'))$ such that $\mathcal{O}_{SK}(M', \rho')$ is already queried upon for the access structure (M', ρ').
8. The returned second level ciphertext is $D = \langle D_0, D_1, D_2, D_3, D_4, D_5 \rangle$, such that the ciphertext component $D_4 = \hat{e}(g,h)^{\alpha(s^*_1 + \beta)} \cdot \hat{e}(g, g_1)^{(s^*_1 + \beta)\alpha d}$.
9. $\mathcal{A}$ parses $D_5 = \langle C'_0, C'_1, C'_2, C'_3, \{C'_x\}, W' \rangle$. Since $SK_{(M', \rho')}$ is known to $\mathcal{A}$, it computes the set of reconstruction constants $\{\omega'_i : i \in I\}$ where $I = \{i \in [l] : att_{\rho(i)'} \in W'\}$ by invoking Reconstruct (M', ρ', W') such that $\sum_{\rho'(i) \in W'} \omega'_i M'_i = 1$ holds and computes $CT_2 = \frac{\hat{e}(C'_1, \prod_{i \in I} K_{1i}^{\omega'_i})}{\prod_{i \in I} \hat{e}(K_{2i}, C'^{\omega'_i}_{\rho(i)})} = \hat{e}(g,h)^{s_1 \alpha}$.

10. It extracts $g^{\alpha d} = D_0/CT_2$ and computes $CT_3 = \hat{e}(g^{\alpha d}, D_2) = \hat{e}(g, g_1)^{(s_1^* + \beta)\alpha d}$.
11. It computes $R^* = D_0 \cdot CT_3/D_4$ and $r^* = H_2(R^*)$.
12. Finally, $\mathcal{A}$ computes $m_\psi = r \oplus C_3^*$.

Note that, as per the security definition of RCCA-secure PRE by Libert *et al.* [17], the adversary cannot issue decryption queries for any second level ciphertext D if $\textbf{Decrypt}(D, SK_{(M',\rho')}) \in \{m_0, m_1\}$ (such an adversarial constraint is not imposed in the security game of CCA secure PRE [10], and the adversary is allowed to issue such decryption queries). Accordingly, the adversary $\mathcal{A}$ does not query the decryption of any challenge ciphertext derivative in the above attack. In fact, the ciphertext component C_3'' is picked at random. The RCCA attack could be launched due to the absence of ciphertext validation in the re-encryption algorithm of the given construction. $\mathcal{A}$ can recover the original message m_ψ encrypted by the challenger towards the target access structure (M^*, ρ^*) as a first level challenge ciphertext C^*, which successfully breaks the RCCA security of the scheme. This completes the description of the attack.

7 Our Unidirectional CCA-secure KP-ABPRE Scheme

7.1 Technical Overview of Construction

The starting points of our construction are the KP-ABE scheme of Rao *et al.* [21] which relies on the threshold public key encryption framework of Qin *et al.* [20] to design the basic construction realising monotone LSSS access structure and BLS short signature [5]. Constant size first-level ciphertexts are achieved by increasing the private key size by a factor of $|W|$, where W is the set of distinct attributes associated with the access structure embedded in the private key. The CA first chooses a random exponent $\alpha \in \mathbb{Z}_p^*$ as the master secret key and computes the public component $Y = \hat{e}(g, g)^\alpha$. The first-level ciphertext of a message $m \in \mathcal{M}$ and an attribute set W is generated using the "Hashed El-Gamal" [8,9] encryption system. First, a random string $\sigma \in \{0,1\}^{l_\sigma}$ is chosen and $s = \mathcal{H}_1(m||\sigma)$ is computed. Then the ciphertext components $C_0 = (m||\sigma) \oplus \mathcal{H}_2(Y^s)$, $C_1 = g^s$, $C_2 = g_1^s$ and $C_3 = \left(h_0 \prod_{att_y \in W} h_y\right)^s$ are computed. Component $C_4 = \left(\mathcal{H}_3(W, C_0, C_1, C_2, C_3)\right)^s$ can be viewed as a BLS signature signing the components (C_0, C_1, C_2, C_3, W) used during decryption/re-encryption to check well-formedness of the first-level ciphertexts. $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{H}_3$ are cryptographic hash functions defined in our construction. Finally, $C = \langle C_0, C_1, C_2, C_3, C_4, W \rangle$ is returned as the first-level ciphertext.

At the second-level, constant size ciphertexts is achieved by increasing the re-encryption key size by a factor of $|W|$, where W is the set of distinct attributes associated with the access structure (M, ρ) embedded in the private key of the delegator. To delegate decryption rights towards an access structure (M', ρ'), the delegator chooses strings $\delta \in \{0,1\}^{l_m}$, $\gamma \in \{0,1\}^{l_\sigma}$, picks an attribute set $W' \models (M', \rho')$ and computes $s' = \mathcal{H}_1(\delta||\gamma)$. Next, it computes re-encryption key components $rk_4 = (\delta||\gamma) \oplus \mathcal{H}_2(Y^{s'})$, $rk_5 = g^{s'}$, $rk_6 = \left(h_0 \prod_{att_y \in W'} h_y\right)^{s'}$ and computes a BLS signature $rk_7 = \big(\mathcal{H}_5(W', rk_4, rk_5, rk_6)\big)^{s'}$ on the components (rk_4, rk_5, rk_6, W'). We construct our re-encryption key in such a way that the string $(\delta||\gamma)$ is blinded with a random salt and can only be removed by a delegatee with private key $SK_{(M',\rho')}$ such that $W' \models (M', \rho')$ during decryption of the re-encrypted ciphertexts. The private key K_i of delegator can be retrieved from the re-encryption key component rk_{1i} only by users with the knowledge random element θ (chosen by delegator). This clearly makes it infeasible to retrieve the private key of the delegator from the re-encryption key and provides collusion-resistance. The CCA-security of the second-level ciphertext follows from the two integrated "hashed" CCA-secure El-Gamal encryptions (D_1, D_3) in the second-level ciphertext $D = (D_0, D_1, D_2, D_3, D_4, D_5, D_6, W')$.

7.2 Our Construction

- **Setup**$(1^\kappa, U)$: The algorithm chooses two bilinear groups $\mathbb{G}_0, \mathbb{G}_1$ of prime order p. Let g and g_1 be generators of $\mathbb{G}_0$, and $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ denote an admissible bilinear map. It chooses a random exponent $\alpha \in \mathbb{Z}_p^*$ and computes $Y = \hat{e}(g, g)^\alpha$. It picks $h_0 \in \mathbb{G}_0$ and for every attribute $att_y \in U$, picks $h_y \in \mathbb{G}_0$. l_m and l_σ are parameters determined by κ, $\{0,1\}^{l_m}$ is the size of the message space $\mathcal{M}$. Let $|U| = n$ be the attribute universe size. Five cryptographic hash functions, modelled as random oracles in the security proof are chosen as follows: $\mathcal{H}_1 : \{0,1\}^{l_m + l_\sigma} \to \mathbb{Z}_p^*, \mathcal{H}_2 : \mathbb{G}_1 \to \{0,1\}^{l_m + l_\sigma}, \mathcal{H}_3 : \{0,1\}^* \times \mathbb{G}_1^3 \to \mathbb{G}_0, \mathcal{H}_4 : \{0,1\}^{l_m} \to \mathbb{Z}_p^*, \mathcal{H}_5 : \{0,1\}^* \times \mathbb{G}_1^2 \to \mathbb{G}_0$. The public parameters returned are $params = \langle \mathbb{G}_0, \mathbb{G}_1, \hat{e}, p, g, g_1, h_0, h_1, \cdots h_n, Y, \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4, \mathcal{H}_5 \rangle$. The master secret key MSK is α.
- **KeyGen**$(MSK, (M, \rho), params)$: On input of an access structure (M, ρ) where $l \times k$ is the size of matrix M, the CA executes Share(M, ρ, α) to obtain a set of l shares $\lambda_{\rho_i} = M_i \cdot v$, where $v \in_R \mathbb{Z}_p^k$, such that $v \cdot \mathbf{1} = \alpha$. Note that $\mathbf{1} = (1, 0, \cdots, 0)$ is a vector of length k. For each row M_i of the matrix M, it picks $r_i \in \mathbb{Z}_p^*$ and computes:

$$K_i = g^{\lambda_{\rho(i)}}(h_0 h_{\rho(i)})^{r_i}, K_i' = g^{r_i},\ K_i'' = \{K_{iy}'' : K_{iy}'' = h_y^{r_i}, \forall y \in [n] \backslash \{\rho(i)\}\}.$$

  It returns $SK_{(M,\rho)} = \langle (M, \rho), \{\forall i \in [l] : K_i, K_i', K_i''\} \rangle$ to the user.
- **Encrypt**$(m, W, params)$: Given as input a message $m \in \mathcal{M}$ and an attribute set W, the first-level encryption algorithm encrypts m as below:
  - Select $\sigma \in \{0,1\}^{l_\sigma}$.
  - Compute $s = \mathcal{H}_1(m||\sigma)$.
  - Compute $C_0 = (m||\sigma) \oplus \mathcal{H}_2(Y^s)$.
  - Compute $C_1 = g^s$, $C_2 = g_1^s$.
  - Compute $C_3 = \big(h_0 \prod_{att_y \in W} h_y\big)^s$.
  - Compute $C_4 = \big(\mathcal{H}_3(W, C_0, C_1, C_2, C_3)\big)^s$.
  - Return the first-level ciphertext $C = \langle C_0, C_1, C_2, C_3, C_4, W \rangle$.
- **Decrypt**$(C, SK_{(M,\rho)}, params)$: On input of a first level ciphertext C and a private key $SK_{(M,\rho)}$, the decryption algorithm works as below:

1. First check if the ciphertext is well-formed as below:

$$\hat{e}(g, C_2) \overset{?}{=} \hat{e}(g_1, C_1) \tag{1}$$

$$\hat{e}(g, C_3) \overset{?}{=} \hat{e}(C_1, h_0 \prod_{att_y \in W} h_y), \tag{2}$$

$$\hat{e}(g, C_4) \overset{?}{=} \hat{e}(C_1, \mathcal{H}_3(W, C_0, C_1, C_2, C_3)). \tag{3}$$

2. If the checks fail, output $\perp$ and abort.
3. Otherwise, execute Reconstruct (M, ρ, W) to obtain a set $\{\omega_i : i \in I\}$ of secret reconstruction constants where $I = \{i \in [l] : att_{\rho(i)} \in W\}$. If $W \models (M, \rho)$, then the relation $\sum_{i \in I} \omega_i \lambda_{\rho(i)} = \alpha$ implicitly holds.
4. Compute: $E_1 = \prod_{i \in I} \left(K_i \cdot \prod_{att_y \in W, y \neq \rho(i)} K''_{i,y} \right)^{\omega_i}$, $E_2 = \prod_{i \in I} (K'_i)^{\omega_i}$
5. Compute the plaintext:

$$(m||\sigma) = C_0 \oplus \mathcal{H}_2 \left(\frac{\hat{e}(C_1, E_1)}{\hat{e}(C_3, E_2)} \right). \tag{4}$$

6. If $C_1 \overset{?}{=} g^{\mathcal{H}_1(m||\sigma)}$, return the plaintext m, else return $\perp$.

- **ReKeyGen**$(SK_{(M,\rho)}, (M, \rho), (M', \rho'), params)$: To generate a re-encryption key from an access structure (M, ρ) to (M', ρ'), the re-encryption key generation algorithm computes the re-key $RK_{(M,\rho) \to (M',\rho')}$ as follows:
  1. Choose $\theta \in \mathbb{Z}_p$. Pick $\delta \in \{0,1\}^{l_m}$ and $\gamma \in \{0,1\}^{l_\sigma}$. For each row M_i of the matrix M of size $l \times k$, compute:

$$rk_{1i} = K_i^{\mathcal{H}_4(\delta)} \cdot \left(h_0 h_{\rho(i)}\right)^{\theta}, rk_{2i} = (K'_i)^{\mathcal{H}_4(\delta)} \cdot g^{\theta},$$
$$rk_{3i} = \{rk_{3iy} : rk_{3iy} = \left(K''^{H_4(\delta)}_{iy} \cdot h_y^{\theta}\right), \forall y \in [n] \backslash \{\rho(i)\}\}.$$

  2. Compute $s' = \mathcal{H}_1(\delta||\gamma)$.
  3. Compute $rk_4 = (\delta||\gamma) \oplus \mathcal{H}_2(Y^{s'})$.
  4. Compute $rk_5 = g^{s'}$.
  5. Pick an attribute set $W' \models (M', \rho')$.
  6. Compute $rk_6 = \left(h_0 \prod_{att_y \in W'} h_y\right)^{s'}$.
  7. Compute $rk_7 = \left(\mathcal{H}_5(W', rk_4, rk_5, rk_6)\right)^{s'}$.
  8. Return the re-encryption key $RK_{(M,\rho) \to (M',\rho')} = (\{\forall i \in [l] : rk_{1i}, rk_{2i}, rk_{3i}\}, rk_4, rk_5, rk_6, rk_7, W')$.
- **ReEncrypt**$(C, RK_{(M,\rho) \to (M',\rho')}, params)$: Given as input a first-level ciphertext C and a re-encryption key $RK_{(M,\rho) \to (M',\rho')}$, the re-encryption algorithm re-encrypts the first-level ciphertext as below:
  1. It checks the validity of the ciphertext C using Eqs. 1, 2, 3.
  2. Check the validity of the re-encryption key by checking if the following equations hold:

$$\hat{e}(g, rk_6) \overset{?}{=} \hat{e}(h_0 \prod_{att_y \in W'} h_y, rk_5) \tag{5}$$

$$\hat{e}(rk_7, g) \overset{?}{=} \hat{e}(\mathcal{H}_5(W', rk_4, rk_5, rk_6), rk_5). \tag{6}$$

  If the above checks fail, return $\perp$.

3. Else if $W \models (M, \rho)$, compute a set $\{\omega_i : i \in I\}$ of secret reconstruction constants where $I = \{i \in [l] : att_{\rho(i)} \in W\}$ using Reconstruct (M, ρ, W) such that $\sum_{\rho(i) \in S} \omega_i M_i = 1$ implicitly holds. Compute $re_1 = \prod_{i \in I} \Big(rk_{1i} \cdot \prod_{att_y \in W, y \neq \rho(i)} rk_{3i} \Big)^{\omega_i}, re_2 = \prod_{i \in I} (rk_{2i})^{\omega_i}$
4. Next, compute:

$$D_0 = \frac{\hat{e}(C_1, re_1)}{\hat{e}(C_3, re_2)} = Y^{s\mathcal{H}_4(\delta)}. \tag{7}$$

5. Set ciphertext components: $D_1 = C_0$, $D_2 = C_1$, $D_3 = rk_4$, $D_4 = rk_5$, $D_5 = rk_6$, $D_6 = rk_7$. Return second level ciphertext $D = \langle D_0, D_1, D_2, D_3, D_4, D_5, D_6, W' \rangle$.

- **ReDecrypt**$(D, SK_{(M', \rho')}, params)$: In order to decrypt a second-level ciphertext D, the decryption algorithm proceeds as below:
  1. Check if the ciphertext is well-formed by checking below equations:

$$\hat{e}(D_4, h_0 \prod_{att_y \in W'} h_y) \stackrel{?}{=} \hat{e}(g, D_5) \tag{8}$$

$$\hat{e}(D_6, g) \stackrel{?}{=} \hat{e}(\mathcal{H}_5(W', D_3, D_4, D_5), D_4). \tag{9}$$

  2. If the check fails, abort and return $\perp$.
  3. Otherwise, if $W' \models (M', \rho')$, compute the set of reconstruction constants $\{\omega'_i : i \in I\}$ where $I = \{i \in [l] : att_{\rho(i)'} \in W'\}$ using Reconstruct (M', ρ', W') such that $\sum_{\rho'(i) \in W'} \omega'_i M'_i = 1$ holds implicitly. Compute:

$$E'_1 = \prod_{i \in I} \Big(K_i \cdot \prod_{att_y \in W', y \neq \rho(i)} K''_{i,y} \Big)^{\omega'_i}, \; E'_2 = \prod_{i \in I} (K'_i)^{\omega'_i}$$

  4. Compute δ as below:

$$(\delta || \gamma) = D_3 \oplus \mathcal{H}_2 \left(\frac{\hat{e}(D_4, E'_1)}{\hat{e}(D_5, E'_2)} \right). \tag{10}$$

  5. If $D_4 \stackrel{?}{=} g^{\mathcal{H}_1(\delta || \gamma)}$ does not hold, return $\perp$. Else, extract plaintext as below:

$$(m || \sigma) = D_1 \oplus \mathcal{H}_2(D_0^{1/\mathcal{H}_4(\delta)}). \tag{11}$$

  6. If $D_2 \stackrel{?}{=} g^{\mathcal{H}_1(m || \sigma)}$, output m, otherwise return $\perp$.

7.3 Correctness

The consistency of our KP-ABPRE scheme is given in the full version of the paper [19] due to space limit.

7.4 Security Proof

Proof Sketch. We now give an intuitive proof sketc.h of selective CCA security of our scheme for both first-level and second-level ciphertexts in the random oracle model, based on the n-Decisional Bilinear Diffie-Hellman Exponentiation (n-DBDHE) assumption. We first analyse the security for first-level ciphertexts. As per the Random Oracle Model, the Challenger handles the adversarial queries as follows: hash queries by oracles $\mathcal{H}_1, \cdots, \mathcal{H}_5$, private key extraction queries by oracle $\mathcal{O}_{SK}(M, \rho)$, re-encryption key generation queries by oracle $\mathcal{O}_{RK}((M, \rho), (M', \rho'))$, re-encryption queries by oracle $\mathcal{O}_{RE}(C, (M, \rho), (M', \rho'))$, decryption queries by oracle $\mathcal{O}_{Dec}(C, (M, \rho))$ and re-decryption queries by oracle $\mathcal{O}_{RD}(D, (M', \rho'))$. In the above oracles, the Challenger injects the hard problem instance into the query responses to Oracles $\mathcal{H}_3$ and private key extraction oracle $\mathcal{O}_{SK}(M, \rho)$ respectively. The first-level and second-level decryption oracles return message m or $\perp$ as per the protocol and the remaining oracles return uniformly random elements from the respective domains.

The Challenger picks $\alpha' \in_R \mathbb{Z}_p^*$ and implicitly sets *msk* as $\alpha = \alpha' + a^{n+1}$, where a^{n+1} is not known to the Challenger. It computes $Y = \hat{e}(g, g)^{\alpha'} \cdot e(g^a, g^{a^n}) = \hat{e}(g, g)^{\alpha}$, as proven in the Initialization Phase of the security proof (Section 7.5, page 20) in the full version of the paper [19]. For an oracle query to $\mathcal{H}_3(W, C_0, C_1, C_2, C_3)$, the Challenger picks $\upsilon \in_R \mathbb{Z}_p^*$ and injects the hard problem instance by computing value $\mu = (g^{a^n})^{\upsilon}$. It returns μ as the value of $\mathcal{H}_3(W, C_0, C_1, C_2, C_3)$. It is easy to follow that the computed value $\mu \in \mathbb{G}_0$ is identically distributed as the real hash value from the construction. For a private key extraction query for access structure (M, ρ), if the target attribute set $W^* \models (M, \rho)$, the Challenger aborts and returns "*failure*". Otherwise, it computes the private keys by appropriately injecting the $2n$ given values of the hard problem instance at applicable points, as discussed in Phase 1 description (Sect. 7.5, page 21) in the full version of the paper [19]. The detailed analysis shows that the keys computed are identically distributed as the keys generated by the KeyGen algorithm in the construction. In the Challenge Phase, the challenger picks $\psi \in \{0, 1\}$ at random and encrypts m_ψ under target attribute set W^* to form challenge ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*, W^*)$ demonstrating that the ciphertext computed is identically distributed as the ciphertext generated by the Encrypt algorithm in the construction. In particular, the hard problem instance Z is injected into the ciphertext component C_0^*, computed as $C_0^* = (m_\psi || \sigma^*) \oplus \mathcal{H}_2(Z \cdot \hat{e}(g^b, g^{\alpha'}))$, where $\sigma^* \in_R \{0, 1\}^{l_\sigma}$. The details are provided in the full version of the paper [19] (Sect. 7.5, page 24). Therefore, once the adversary $\mathcal{A}$ produces its guess $\psi' \in \{0, 1\}$, if $\psi' = \psi$, $\mathcal{A}$ wins the game and the Challenger decides $\hat{e}(g^b, g^{a^{n+1}}) = Z$, else Z is random. The security game for second-level ciphertexts proceeds in a similar way as the first-level security game. The computation of the challenge second level ciphertext $D^* = (D_0^*, D_1^*, D_2^*, D_3^*, D_4^*, D_5^*, D_6^*, W^*)$ shown in the Challenger Phase (Sect. 7.6, page 29) in the full version of the paper [19] indicates that the ciphertext computed is identically distributed as the ciphertext generated by the ReEncrypt algorithm in the construction. In particular, the hard problem instance is injected in the ciphertext component D_3^*, where $D_3^* = (\delta^*||\gamma^*) \oplus \mathcal{H}_2(Z \cdot \hat{e}(g^b, g^{\alpha'})) = (\delta^*||\gamma^*) \oplus \mathcal{H}_2(Y^{s'})$. Therefore, once the adversary $\mathcal{A}$ produces its guess $\psi' \in \{0,1\}$, if $\psi' = \psi$, $\mathcal{A}$ wins the game and the Challenger decides $\hat{e}(g^b, g^{a^{n+1}}) = Z$, else Z is random.

First Level Ciphertext Security

Theorem 1. *If a (t,ϵ)IND-PRE-CCA adversary $\mathcal{A}$ has a non-negligible advantage ϵ in breaking the CCA security of the given KP-ABPRE scheme for first level ciphertext, with access to random oracles $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4, \mathcal{H}_5$, then there exists an algorithm $\mathcal{C}$ that solves the n-DBDHE problem with advantage ϵ' within time t' where,*

$$\epsilon' \geq \frac{1}{q_{\mathcal{H}_2}}\left(2\epsilon - \frac{q_{\mathcal{H}_1}}{2^{l_m+l_\sigma}} - \frac{q_{ReEnc}}{p} - q_{Dec}\left(\frac{q_{\mathcal{H}_1}}{(2^{l_m+l_\sigma})} + \frac{1}{p}\right)\right),$$

$$t' \leq t + (q_{\mathcal{H}_3} + q_{\mathcal{H}_5} + (n+2)lq_{SK} + (n+6)lq_{RK} + 7q_{ReEnc} + 3q_{Dec} + 5q_{RD})t_e + (8q_{ReEnc} + 8q_{Dec} + 6q_{RD})t_{bp}.$$

where l is the number of rows in access structure (M,ρ), n is the size of attribute universe and t_e, t_{bp} denote the time taken for exponentiation and pairing operations respectively.

Proof. Due to space limit, the proof is given in the full version of the paper [19].

Second Level Ciphertext Security

Theorem 2. *If a (t,ϵ)IND-PRE-CCA adversary $\mathcal{A}$ has a non-negligible advantage ϵ in breaking the IND-PRE-CCA security of the given KP-ABPRE scheme for second level ciphertext, with access to the random oracles $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4, \mathcal{H}_5$, then there exists an algorithm $\mathcal{C}$ that solves the n-DBDHE problem with advantage ϵ' within time t' where,*

$$\epsilon' \geq \frac{1}{q_{\mathcal{H}_2}}\left(2\epsilon \cdot \left(1 - \frac{q_{\mathcal{H}_4}}{2^{l_m}}\right) - \frac{q_{\mathcal{H}_1}}{2^{l_m+l_\sigma}} - q_{Dec}\left(\frac{q_{\mathcal{H}_1}}{(2^{l_m+l_\sigma})} + \frac{1}{p}\right)\right),$$

$$t' \leq t + (q_{\mathcal{H}_3} + (n+2)lq_{SK} + (n+6)lq_{RK} + 3q_{Dec} + 5q_{RD})t_e + (8q_{Dec} + 6q_{RD})t_{bp}.$$

Proof. Due to space limit, the proof is given in the full version of the paper [19].

Collusion Resistance

Theorem 3 *[16]. If a unidirectional single-hop KP-ABPRE scheme is IND-PRE-CCA secure for first level ciphertexts, then it is collusion resistant as well.*

8 Conclusion

Although several KP-ABPRE schemes have been proposed in the literature, to the best of our knowledge, only one CCA-secure scheme due to Ge *et al.* has reported the collusion resistant property. In this paper, we demonstrate a CCA-attack on their construction. We also give a construction of the first unidirectional KP-ABPRE scheme with constant size ciphertexts requiring a constant number of exponentiations in encryption, and constant number of bilinear pairing operations during decryption and re-decryption while satisfying selective CCA-security for first-level and second level ciphertexts. Also, the definition of collusion resistance is met wherein a colluding proxy and delegatee cannot obtain the private key of the delegator. Our work affirmatively resolves the problem of collusion resistance by proposing a computationally efficient collusion resistant KP-ABPRE scheme that supports monotonic access structures for fine-grained delegation of decryption rights.

References

1. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. In: NDSS (2005)
2. Beimel, A.: Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of technology, Technion, Haifa, Israel (1996)
3. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20–23 May 2007, California, USA, pp. 321–334. IEEE Computer Society (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). https://doi.org/10.1007/BFb0054122
5. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. J. Cryptol. **17**(4), 297–319 (2004)
6. Chung, P.-S., Liu, C.-W., Hwang, M.-S.: A study of attribute-based proxy re-encryption scheme in cloud environments. I. J. Netw. Secur. **16**(1), 1–13 (2014)
7. Cohen, A.: What about bob? the inadequacy of CPA security for proxy reencryption. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 287–316. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_10
8. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptol. **26**(1), 80–101 (2013)
9. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_2
10. Ge, C., Susilo, W., Fang, L., Wang, J., Shi, Y.: A CCA-secure key-policy attribute-based proxy re-encryption in the adaptive corruption model for dropbox data sharing system. Des. Codes Crypt. **86**(11), 2587–2603 (2018)
11. Ge, C., Susilo, W., Wang, J., Huang, Z., Fang, L., Ren, Y.: A key-policy attribute-based proxy re-encryption without random oracles. Comput. J. **59**(7), 970–982 (2016)
12. Li, K., Wang, J., Zhang, Y., Ma, H.: Key policy attribute-based proxy re-encryption and RCCA secure scheme. J. Internet Serv. Inf. Secur. **4**(2), 70–82 (2014)

13. Li, K., Zhang, Y., Ma, H.: Key policy attribute-based proxy re-encryption with matrix access structure. In: 2013 5th International Conference on Intelligent Networking and Collaborative Systems, Xi'an, China, 9–11 September 2013, pp. 46–50 (2013)
14. Liang, K., Fang, L., Wong, D.S., Susilo, W.: A ciphertext-policy attribute-based proxy re-encryption scheme for data sharing in public clouds. Concurrency Comput. Pract. Exper. **27**(8), 2004–2027 (2015)
15. 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, pp. 276–286. ACM (2009)
16. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 360–379. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78440-1_21
17. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. IEEE Trans. Inf. Theor. **57**(3), 1786–1802 (2011)
18. Luo, S., Hu, J., Chen, Z.: Ciphertext policy attribute-based proxy re-encryption. In: Soriano, M., Qing, S., López, J. (eds.) ICICS 2010. LNCS, vol. 6476, pp. 401–415. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17650-0_28
19. Paul, A., Selvi, S.S.D., Rangan, C.P.: Attribute-based proxy re-encryption with constant size ciphertexts. IACR Cryptol. ePrint Arch., 2019:1325 (2019). https://eprint.iacr.org/2019/1325/20201021:184704
20. Qin, B., Qianhong, W., Zhang, L., Farràs, O., Domingo-Ferrer, J.: Provably secure threshold public-key encryption with adaptive security and short ciphertexts. Inf. Sci. **210**, 67–80 (2012)
21. Rao, Y.S., Dutta, R.: Computational friendly attribute-based encryptions with short ciphertext. Theor. Comput. Sci. **668**, 1–26 (2017)
22. Sahai, A., Waters, B.:. Fuzzy identity-based encryption. In: Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005, Proceedings, pp. 457–473. Springer (2005)
23. Weng, J., Deng, R.H., Ding, X., Chu, C.-K., Lai, J.: Conditional proxy re-encryption secure against chosen-ciphertext attack. In: Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2009, Sydney, Australia, 10–12 March 2009, pp. 322–332 (2009)