A Ciphertext-Policy Attribute-Based Proxy Re-Encryption with Chosen-Ciphertext Security

Kaitai Liang[1], Liming Fang[2], Duncan S. Wong[1], and Willy Susilo[3]

[1] Department of Computer Science, City University of Hong Kong, China
kliang4-c@my.cityu.edu.hk, duncan@cityu.edu.hk
[2] Nanjing University of Aeronautics and Astronautics, Jiangsu, China
fangliming@nuaa.edu.cn
[3] School of Computer Science and Software Engineering, University of Wollongong, Australia
wsusilo@uow.edu.au

Abstract. Ciphertext-Policy Attribute-Based Proxy Re-Encryption (CP-ABPRE) extends the traditional Proxy Re-Encryption (PRE) by allowing a semi-trusted proxy to transform a ciphertext under an access policy to the one with the same plaintext under another access policy (i.e. *attribute-based re-encryption*). The proxy, however, learns nothing about the underlying plaintext. CP-ABPRE has many real world applications, such as fine-grained access control in cloud storage systems and medical records sharing among different hospitals. Previous CP-ABPRE schemes leave how to be secure against chosen-ciphertext attacks (CCA) as an open problem. This paper, for the first time, proposes a new CP-ABPRE to tackle the problem. The new scheme supports attribute-based re-encryption with any monotonic access structures. Despite our scheme is constructed in the random oracle model, it can be proved CCA secure under the decisional q-parallel bilinear Diffie-Hellman exponent assumption.

Keywords: Proxy Re-Encryption, Ciphertext-Policy Attribute-Based Proxy Re-Encryption, Bilinear Map, Chosen-Ciphertext Security.

1 Introduction

Introduced by Sahai and Waters [25], Attribute-Based Encryption (ABE), which is a generalization of Identity-Based Encryption (IBE), is able to effectively increase the flexibility of data sharing such that only parties satisfying specific policy are allowed to access the data. It comes in two flavors: one is the key-policy ABE (KP-ABE), and the other is the ciphertext-policy ABE (CP-ABE). In the former, ciphertexts are labeled with attribute sets and private keys are associated with access structures that specify which kinds of ciphertexts the receiver is able to decrypt. In the latter, however, the case is complementary. That is, ciphertexts are related to access structures, and attribute sets are assigned to private keys. ABE is applicable to many network applications, such as targeted broadcast and audit log applications [15].

We use medical data sharing as an example to illustrate the usage of CP-ABE and motivate our work as well. Consider the following scenario. A heart-disease patient Alice would like to find a clinic for regular medical examination via an on-line medical service agent (e.g., healthgrades[4]). The clinic must be located within 10 km of Campbelltown, the doctors (assigned to her) of the clinic must be the senior attending doctors and be expert at cardiology. For convenience, we denote Alice's requirements as $I_1 = \{Cardiology \wedge Senior\ Attending\ Doctor \wedge Location :\ within\ 10\ km\ of\ Campbelltown\}$. To protect the confidentiality of her medical record, Alice prefers to encrypt the record under I_1 (i.e. $Enc_{I_1}(m_{Alice})$) before sending to the on-line medical service agent ("the Agent"). The Agent (that knows I_1) then searches candidates satisfying I_1 in its database. Suppose there is a clinic matching I_1. The Agent forwards Alice's ciphertext to the clinic. Note that the Agent cannot access the medical data without knowledge of the private key (where such a key is associated with an attribute set satisfying I_1).

[4] http://www.healthgrades.com/.

Upon receiving Alice's ciphertext, the clinic that satisfies I_1 is able to decrypt the ciphertext using its private key so as to access the medical record. To keep trace of the medical record, the clinic may back up the ciphertext locally. Suppose some cooperation is required in the process of the treatment, Alice's medical record has to be transferred to hospitals with the following requirements. The hospitals have to be located within 15 km of Hurstville, and the doctors (assigned to the cooperation) of the hospitals should be the attending or chief doctors and must be expert at cardiology as well. Denote the above requirements as $I_2 = \{Cardiology \wedge (Attending\ Doctor \vee Chief\ Doctor) \wedge Location : within\ 15\ km\ of\ Hurstville\}$. Suppose there are two hospitals, say hospital A and hospital B, which satisfy I_2.

In traditional data sharing, sharing Alice's medical record with A and B (without losing confidentiality), the clinic has to first recover m_{Alice} and further encrypt the record under I_2 (i.e. $Enc_{I_2}(m_{Alice})$) before sending to A and B. However, if there are N patients who need to be cooperatively treated among the clinic, A and B, then the clinic will suffer from N pairs of encryption and decryption for their patients' records (See Fig. 1). This might be undesirable in practice due to high computational complexity.

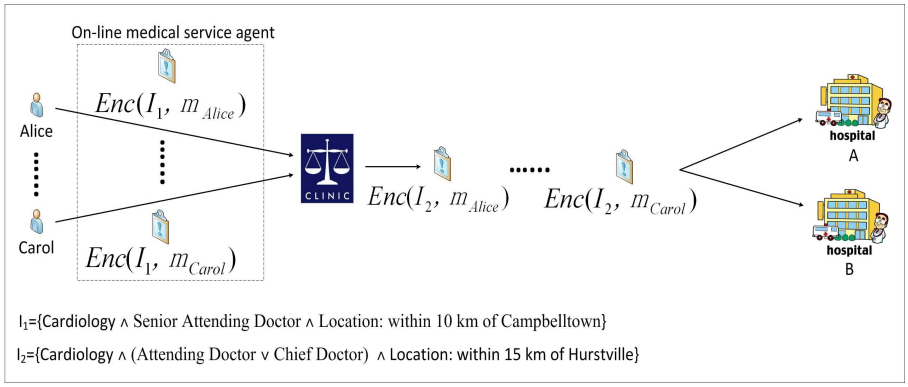

Fig. 1. Traditional Attribute-Based Encryption Data Sharing

To make data sharing be more efficiently, Proxy Re-Encryption (PRE) is proposed. Introduced by Mambo and Okamoto [23] and first defined by Blaze, Bleumer and Strauss [6], PRE extends the traditional Public Key Encryption (PKE) to support the delegation of decryption rights. It allows a semi-trusted party called *proxy* to transform a ciphertext intended for a party A into another ciphertext of the same plaintext intended for party B. The proxy, however, learns neither the decryption keys nor the underlying plaintext. PRE is a useful cryptographic primitive and has many applications, such as secure distributed files systems [1,2] and email forwarding [6].

To date, PRE has been extended to adapt different cryptographic settings. Employing PRE in the context of ABE, Liang et al. [20] proposed the first ciphertext-policy attribute-based proxy re-encryption (CP-ABPRE) scheme, in which a proxy is allowed to transform a ciphertext under a specified access policy into the one under another access policy (i.e. *attribute-based re-encryption*).

Using CP-ABPRE the medical records sharing above can be efficiently fulfilled as follows (See Fig. 2). The clinic, acting as a delegator, notifies a cloud (storage) server (acting as the proxy) that the hospitals satisfying I_2 (i.e., delegatees), should be granted the decryption rights of the medical records. The server will then transform the ciphertexts of the medical records under I_1 to the ones under I_2 using a re-encryption key (e.g., $rk_{I_1 \to I_2}$ which is generated by the clinic), such that A and B (the hospitals satisfying I_2) can decrypt the records. Note that the server does not learn the contents (of the medical records).

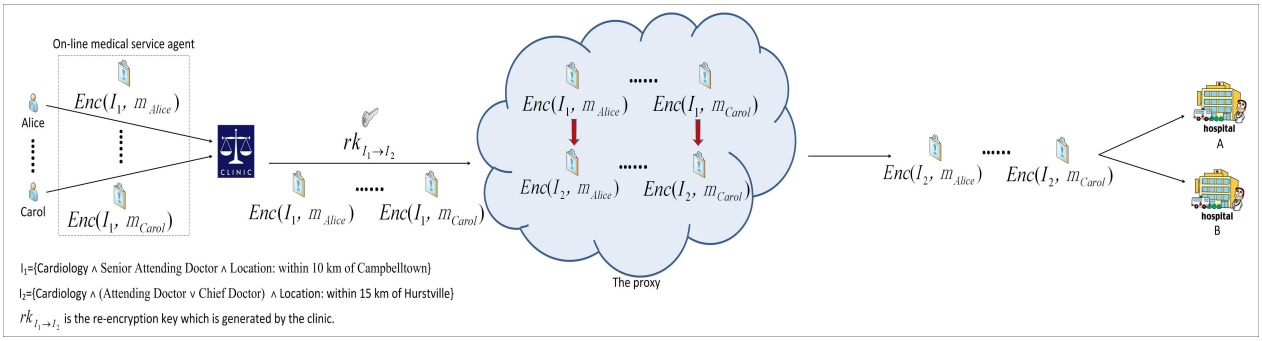

Fig. 2. Ciphertext-policy attribute-based proxy re-encryption

We argue that CP-ABPRE explores the applications of PRE and has many real world applications, such as fine-grained data sharing in on-line medical service systems (such as Healthgrades[5], Scripps Health[6]). For example, in an on-line medical service system, a couple (who settles down in Sydney) prefers to find doctors with the following requirements to remedy their child's bronchitis. Denote the requirements as $I_3 = \{Paediatrician \wedge Bronchitis \wedge (Consultant \vee Registrar) \wedge Location : Downtown\ of\ Sydney\}$. The parent encrypts the child's medical record under I_3 before uploading to the system. Because the system has no corresponding private key related to I_3, it cannot access the record. The system then forwards the ciphertext to the doctors satisfying I_3[7]. Nevertheless, when one of the doctors goes out for medical trip or for vacation, it is necessary to find some trustworthy substitutes whom can check the medical record. By employing CP-ABPRE a doctor can first specify a new access policy, such as $I_4 = \{Paediatrician \wedge Bronchitis \wedge (Senior\ Registrar \vee Registrar)\}$, and then generates a re-encryption key (which can transform the ciphertext under I_3 into the one under I_4) for his/her proxy. When the doctor is absent, the proxy can translate the ciphertext of the record to the one which can be only decrypted by the doctors satisfying I_4.

Previous CP-ABPRE schemes are only secure against *chosen-plaintext attacks* (CPA). The existence of CP-ABPRE with *chosen-ciphertext security* has been open. We note that CPA security might be not sufficient in general protocol settings as it only achieves the very basic requirement from an encryption scheme, i.e. secrecy against "passive" eavesdroppers. When CP-ABPRE is implemented within a large protocol or system, a much wider array of attacks are possible. For example, the adversary may have control over the ciphertexts so as to affect the decryption values or learn some partial information of decryption result.

CCA security, however, allows the adversary to access the decryption oracle, i.e. achieving the ability to read the underlying plaintext related to the ciphertexts of its choice. This is able to preclude insider attacks. For example, a legitimate doctor of some hospital is able to acquire pairs of CP-ABPRE ciphertexts and plaintexts as previous knowledge. But the CCA security guarantees that he/she still cannot gain any useful knowledge of the underlying plaintext of the challenge ciphertext after his/her retirement. CCA security also implies *non-malleability* that guarantees that if the adversary modifies given ciphertexts, then the corresponding decryption yields invalid results. That is, even if the ciphertexts are modified and re-transferred to other hospitals (whom are not the recipients specified by original sender), the underlying medical records still cannot be accessed. Therefore, it is desirable to propose CCA secure CP-ABPRE scheme in practice.

[5] http://www.healthgrades.com/.
[6] http://www.scripps.org/.
[7] Suppose the system uses a cloud (with considerable storage capability) to store the information of doctors.

Another open problem left by previous CP-ABPRE schemes is how to support any monotonic access policy. In practical use, it is desirable to enable a CP-ABPRE to support expressive and flexible realization for access policy. This paper also deals with this problem.

1.1 Our Contributions

In this work we formalize the definition for CP-ABPRE. Specifically, in our definition an attribute set and an access structure is required as auxiliary input to the re-encryption key algorithm; meanwhile, an attribute set is required in the input to the private key generation and decryption algorithms.

Regarding to the security models, we propose the *selective access structure and chosen ciphertext security* (IND-sAS-CCA) game for CP-ABPRE. Note that it is the first time to define chosen ciphertext security model for CP-ABPRE in the literature; and meanwhile, the game above can be easily converted to the *adaptive access structure and chosen ciphertext security* (IND-aAS-CCA) one (details can be seen in Section 2.2). We consider the IND-sAS-CCA game into two different aspects: one is to allow the adversary to achieve an original ciphertext as the challenge ciphertext; the other is to allow the adversary to achieve a re-encrypted ciphertext as challenge. We refer to the security of the former and the latter as *IND-sAS-CCA security at original ciphertext* (i.e. IND-sAS-CCA-Or) and *IND-sAS-CCA security at re-encrypted ciphertext* (i.e. IND-sAS-CCA-Re), respectively. In this paper we also show that the IND-sAS-CCA-Or security implies *selective collusion resistance*. Note that in [22] selective collusion resistance is also called as *selective master key security*.

The construction of a CP-ABPRE with CCA security is an open problem left by previous CP-ABPRE schemes. This paper proposes the first single-hop unidirectional CP-ABPRE to tackle the problem. It is also worth mentioning that the existing CP-ABPRE schemes *only* support AND gates on (multi-valued) positive and negative attributes, while our scheme allows ciphertexts to be associated with any monotonic access formula. Despite our scheme is constructed in the random oracle model, it can be proved IND-sAS-CCA secure under the decisional q-parallel bilinear Diffie-Hellman exponent (q-parallel BDHE) assumption.

Difficulty of Converting Previous CPA Secure CP-ABPRE to Be Secure against CCA. As stated in [20], to convert a CPA secure CP-ABPRE scheme to be CCA one is a challenging open problem. One might think that some cryptographic primitives might come to help, such as the CHK transformation [7]. The well-known CHK transformation can be used to convert a CPA secure PKE scheme to be secure against CCA. The transformation, however, cannot be trivially employed in a CPA secure PRE scheme so as to achieve CCA security. This is so because the CHK transformation is used to prevent ciphertext from being mutated, but at the same time, PRE allows a ciphertext to be transformed into another ciphertext. Namely, we use CHK transformation to guarantee the validity of ciphertexts leads to a fact that the modification part brought by re-encryption cannot be virtually protected. Thus, trivially employ the CHK transformation in the PRE setting often results in a Replayable CCA security (RCCA) [9]. The classic instances are [21,11,13].

We here use an example to make a specific explanation. Suppose there is a CPA secure CP-ABPRE scheme which is constructed in the standard model, and its original ciphertext is (A, B, C). In re-encryption, suppose the proxy at least generates (at least) a new component A', and outputs (A', B, C) as the re-encrypted ciphertext such that the corresponding delegatee can recover the plaintext from (A', B, C) by using his private key. Using the CHK transformation, the delegator may make a signature D for (A, B, C) and output (K_v, A, B, C, D) as the original ciphertext, where K_v is the verification key corresponding to D. To validate the signature, the proxy has to keep A as an auxiliary output, i.e. outputting (K_v, A, A', B, C, D). Despite the integrity of (A, B, C) can be verified by K_v and D, A' can be arbitrarily mutated by adversary. On the other hand, if only B and C are bound by D, then A's integrity cannot be guaranteed. Note that to validate the CHK transformation one keynote should be noticed that the verification key K_v must be "sealed" in ciphertext components such that

an adversary cannot simply choose a new signing and verification key pair (K'_v, K'_s) and further make a new signature D' for (A, B, C).

One might doubt that the proxy might choose to sign A' such that the re-encrypted ciphertext is bound by signature. Nevertheless, this approach seems to be insensible. Suppose the proxy makes a signature D' for A' using a new signing key K'_s, and outputs $(K_v, A, B, C, D, K'_v, A', D')$ as the re-encrypted ciphertext, where K'_v is the verification key associated with D'. An adversary may launch the following attacks: it first mutates A' as A'', next chooses (K''_s, K''_v), and then signs A'' in D''. The adversary finally outputs $(K_v, A, B, C, D, K''_v, A'', D'')$. Here the verification is passed but the ciphertext is mutated. The reason behind the attacks is that A' as a single component is loosely related to the original ciphertext and K'_v.

A naive solution for the problem is to request the proxy to not only encrypt A' under a new access policy which associates with the delegatees' attributes, but also make signature for the new ciphertext as the manner of original ciphertext. However, this approach comes at a price that the communication bandwidth and decryption complexity are both increased. Furthermore, the proxy might suffer from malicious attacks or invasion by adversary. Thus, this solution might be easily experienced the same attacks introduced in the previous paragraph. In some privacy-preserving CP-ABPRE setting, i.e. the proxy does not know the corresponding delegatees' attributes, this solution is inappropriate as well.

Therefore, using the CHK transformation as a black box to turn the existing CPA secure CP-ABPRE schemes to be secure against CCA that is not trivial. In section 4, we show an efficient solution to address the difficulty.

1.2 Related Work

In 2005 Sahai and Waters [25] introduced the concept of ABE. There are two categories of ABE, KP-ABE and CP-ABE. Goyal et al. [15] proposed the first KP-ABE, in which a ciphertext is related to a set of attributes, and each private key corresponds to an access policy over the attributes. The decryption can be fulfilled correctly if and only if the attribute set of the ciphertext satisfies the access policy on the decryptor's private key. Reversely, Bethencourt et al. [5] proposed CP-ABE where the ciphertext is associated with an access policy and the private key is related to an attribute set. Note that we here mainly focus on reviewing CP-ABE. Later on, Cheung and Newport [10] proposed a provably secure CP-ABE scheme which only supports AND gates over attributes. The first fully expressive CP-ABE was proposed by Waters [26]. Using dual system encryption, Lewko et al. [18] proposed a fully secure CP-ABE which leads to some loss of efficiency compared to the most efficient scheme proposed in [26]. Recently, Attrapadung et al. [3] proposed a CP-ABE with constant-size ciphertexts.

Following the introduction of decryption rights delegation by Mambo and Okamoto [23], Blaze et al. [6] formalized proxy re-encryption and proposed a seminal bidirectional PRE scheme. After that, Ivan and Dodis [17] formalized the definitions of bidirectional and unidirectional proxy functions. In 2005, Ateniese et al. [1,2] proposed three unidirectional PRE schemes with CPA security. Later on, many classic PRE schemes (e.g., [8,16,21]) have been proposed.

To implement PRE in the attribute-based cryptographic setting, Liang et al. [20] defined CP-ABPRE, and proposed a concrete construction based on a CP-ABE scheme [10] in which access policy is only represented as AND gates on positive and negative attributes. Mizuno and Doi [24] proposed a hybrid PRE scheme (in general) where the scheme can bridge ABE and IBE in the sense that ciphertexts generated in the context of ABE can be converted to the ones which can be decrypted in the IBE setting. Luo et al. [22] proposed a CP-ABPRE scheme which supports AND gates on multi-valued and negative attributes. The aforementioned CP-ABPRE schemes, however, are only secure against CPA and supports AND gates over attributes. The construction of a CCA secure CP-ABPRE supporting any monotonic access policy remains open. This paper deals with this problem.

We here compare our scheme with previous CP-ABPRE schemes, and summarize the comparison in terms of public/private key size, ciphertext/re-encryption key size, re-encryption cost and properties,

in Table 1. We let f be the size of an access formula, A be the number of attributes on a user's private key, U be the number of all attributes defined in the system, mv be multi-valued attribute, $+$ be positive attribute and $-$ be negative attribute. Besides, we use c_e and c_p to denote the computational cost of an exponetiation and a bilinear pairing. To the best of our knowledge, our scheme is the first of its kind to achieve CCA security and to support any monotonic access formula (over attributes).

Table 1. Comparison with [20,22,24]

Schemes	Public/Private Key Size	Ciphertext/ ReKey Size	Re-Encryption Cost	Selective Model /CCA Security	Attributes Expression
CP-ABPRE [20]	$\mathcal{O}(U)/\mathcal{O}(U)$	$\mathcal{O}(U)/\mathcal{O}(U)$	$\mathcal{O}(U)\cdot c_p$	✓ /✗	AND gates on $+$ and $-$
CP-ABPRE [22]	$\mathcal{O}(U^2)/\mathcal{O}(U)$	$\mathcal{O}(U)/\mathcal{O}(U)$	$\mathcal{O}(U)\cdot c_p$	✓ /✗	AND gates on mv and $-$
CP-ABPRE [24]	$\mathcal{O}(U)/\mathcal{O}(U)$	$\mathcal{O}(U)/\mathcal{O}(U)$	$\mathcal{O}(1)\cdot c_e+\mathcal{O}(U)\cdot c_p$	✓ /✗	AND gates on $+$ and $-$
Our CP-ABPRE	$\mathcal{O}(1)/\mathcal{O}(A)$	$\mathcal{O}(f)/\mathcal{O}(A)$	$\mathcal{O}(A)\cdot c_e+\mathcal{O}(A)\cdot c_p$	✓ /✓	Any monotonic access formula

2 Definitions and Security Models

In this section, we concentrate on formulating the definition of CP-ABPRE systems. Before proceeding, we first review some notations used in our definition.

Definition 1. *(**Access Structure** [4]) Let $\mathcal{P}=\{P_1,P_2,...,P_n\}$ be a set of parties. A collection $\mathbb{AS}\subseteq 2^{\mathcal{P}}$ is monotone if $\forall B,C$: if $B\in\mathbb{AS}$ and $B\subseteq C$ then $C\in\mathbb{AS}$. An access structure (resp., monotonic access structure) is a collection (resp., monotone collection) $\mathbb{AS}$ of non-empty subsets of $\mathcal{P}$, i.e., $\mathbb{AS}\subseteq 2^{\mathcal{P}}\setminus\{\emptyset\}$. The sets in $\mathbb{AS}$ are called the authorized sets, and the sets not in $\mathbb{AS}$ are called the unauthorized sets.*

In ABE the role of the parties is taken by the attributes. The access structure $\mathbb{AS}$ contains all authorized sets of attributes. In this paper we work on monotone access structures. As shown in [4], any monotone access structure can be represented by a linear secret sharing scheme.

Definition 2. *(**Linear Secret Sharing Schemes (LSSS)** [26]) A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if*

- *The shares for each party form a vector over $\mathbb{Z}_p$.*
- *There exists a matrix an M with l rows and n columns called the share-generating matrix for Π. For all $i=1,...,l$, the ith row of M is labeled by a party $\rho(i)$, where ρ is a function from $\{1,...,l\}$ to $\mathcal{P}$. When we consider the column vector $v=(s,r_2,...,r_n)$, where $s\in\mathbb{Z}_p$ is the secret to be shared, and $r_2,...,r_n\in\mathbb{Z}_p$ are randomly chosen, then $M\cdot v$ is the vector of l shares of the secret s according to Π. The share $(M\cdot v)_i$ belongs to party $\rho(i)$. For any unauthorized set, no such constants exist. We use LSSS matrix (M,ρ) to represent an access policy in this paper.*

Note that every LSSS according to the above definition achieves the *linear reconstruction* property [4]. Suppose that Π is an LSSS for the access structure $\mathbb{AS}$. Let $S\in\mathbb{AS}$ (that is, S satisfies the access structure; we also denote this case as $S\models(M,\rho)$) be any authorized set, and let $I\subset\{1,2,...,l\}$ be defined as $I=\{i:\rho(i)\in S\}$. There will exist constants $\{w_i\in\mathbb{Z}_p\}_{i\in I}$ such that $\sum_{i\in I}w_i\cdot\lambda_i=s$ if $\{\lambda_i\}$ are valid shares of any secret s according to Π. Note that as shown in [4] $\{w_i\}$ can be found (with knowledge of M and I) in time polynomial in the size of the share-generating matrix M.

2.1 Definition of CP-ABPRE

Definition 3. *A Single-Hop Unidirectional Ciphertext-Policy Attribute-Based Proxy Re-Encryption (CP-ABPRE) scheme consists of the following seven algorithms:*

1. $(param, msk) \leftarrow Setup(1^k, \mathcal{U})$*: on input a security parameter* $k \in \mathbb{N}$ *and an attribute universe* $\mathcal{U}$*, output the public parameters* $param$ *and a master secret key* msk*.*
2. $sk_S \leftarrow KeyGen(param, msk, S)$*: on input* $param$*,* msk *and an attribute set* S *that describes the key, output a private key* sk_S *for* S*. Note that like traditional CP-ABE each private key* sk_S *is associated with an attribute set* S*.*
3. $rk_{S\rightarrow(M',\rho')} \leftarrow ReKeyGen(param, sk_S, S, (M', \rho'))$*: on input* $param$*, a private key* sk_S *and the corresponding attribute set* S*, and an access structure* (M', ρ') *for attributes over* $\mathcal{U}$*, output a re-encryption key* $rk_{S\rightarrow(M',\rho')}$ *that can be used to transform a ciphertext under* (M, ρ) *to another ciphertext under* (M', ρ')*, where* $S \models (M, \rho)$*. Note that* (M, ρ) *and* (M', ρ') *are disjoint*[8]*.*
4. $C_{(M,\rho)} \leftarrow Enc(param, (M, \rho), m)$*: on input* $param$*, an access structure* (M, ρ) *for attributes over* $\mathcal{U}$*, and a plaintext* $m \in \{0,1\}^k$*, output an original ciphertext* $C_{(M,\rho)}$ *which can be further re-encrypted. We assume that the access structure is implicitly included in the ciphertext.*
5. $C^R_{(M',\rho')} \leftarrow ReEnc(param, rk_{S\rightarrow(M',\rho')}, C_{(M,\rho)})$*: on input* $param$*, a re-encryption key* $rk_{S\rightarrow(M',\rho')}$*, and an original ciphertext* $C_{(M,\rho)}$*, output a re-encrypted ciphertext* $C^R_{(M',\rho')}$ *if* $S \models (M, \rho)$ *or a symbol* $\perp$ *indicating either* $C_{(M,\rho)}$ *is invalid or* $S \not\models (M, \rho)$*. Note that* $C^R_{(M',\rho')}$ *cannot be further re-encrypted.*
6. $m \leftarrow Dec(param, S, sk_S, C_{(M,\rho)})$*: on input* $param$*, an attribute set* S *and its corresponding private key* sk_S*, and an original ciphertext* $C_{(M,\rho)}$*, output a plaintext* m *if* $S \models (M, \rho)$ *or a symbol* $\perp$ *indicating either* $C_{(M,\rho)}$ *is invalid or* $S \not\models (M, \rho)$*.*
7. $m \leftarrow Dec_R(param, S', sk_{S'}, C^R_{(M',\rho')})$*: on input* $param$*, an attribute set* S' *and its corresponding private key* $sk_{S'}$*, and a re-encrypted ciphertext* $C^R_{(M',\rho')}$*, output a plaintext* m *if* $S' \models (M', \rho')$ *or a symbol* $\perp$ *indicating either* $C^R_{(M',\rho')}$ *is invalid or* $S' \not\models (M', \rho')$*.*

For simplicity, we omit $param$ in the expression of the algorithm inputs in the rest of the paper.

Correctness: For any $k \in \mathbb{N}$, any attribute set S ($S \subseteq \mathcal{U}$) with its cardinality polynomial to k, any access structure (M, ρ) for attributes over $\mathcal{U}$ and any message $m \in \{0,1\}^k$, if $(param, msk) \leftarrow Setup(1^k, \mathcal{U})$, $sk_S \leftarrow KeyGen(msk, S)$, for all S used in the system, we have

$$Dec(S, sk_S, Enc((M, \rho), m)) = m;$$
$$Dec_R(S', sk_{S'}, ReEnc(ReKeyGen(sk_S, S, (M', \rho')), Enc((M, \rho), m))) = m,$$

where $S \models (M, \rho)$ and $S' \models (M', \rho')$.

2.2 Security Models

In the following we define the security notions for CP-ABPRE systems. Prior models for CP-ABPRE only consider the IND-sAS-CPA security, below we define a complete IND-sAS-CCA security game.

Definition 4. *A single-hop unidirectional CP-ABPRE scheme is IND-sAS-CCA secure at original ciphertext if no probabilistic polynomial time (PPT) adversary* $\mathcal{A}$ *can win the game below with non-negligible advantage. In the game,* $\mathcal{C}$ *is the game challenger,* k *and* $\mathcal{U}$ *are the security parameter and attribute universe.*

1. ***Initialization.*** $\mathcal{A}$ *outputs a challenge access structure* (M^*, ρ^*) *to* $\mathcal{C}$*.*

[8] Suppose (M, ρ) and (M', ρ') are two access structures. For any attribute x satisfies (M, ρ), x does not satisfy (M', ρ'). For such a case, from now on, we say that (M, ρ) and (M', ρ') are disjoint.

2. ***Setup.*** *$\mathcal{C}$ runs $Setup(1^k, \mathcal{U})$ and sends param to $\mathcal{A}$.*
3. ***Query Phase I.*** *$\mathcal{A}$ is given access to the following oracles.*
   (a) *Private key extraction oracle $\mathcal{O}_{sk}(S)$: on input an attribute set S, $\mathcal{C}$ runs $sk_S \leftarrow KeyGen(msk, S)$ and returns sk_S to $\mathcal{A}$.*
   (b) *Re-encryption key extraction oracle $\mathcal{O}_{rk}(S, (M', \rho'))$: on input an attribute set S, and an access structure (M', ρ'), $\mathcal{C}$ returns $rk_{S\rightarrow(M',\rho')} \leftarrow ReKeyGen(sk_S, S, (M', \rho'))$ to $\mathcal{A}$, where $sk_S \leftarrow KeyGen(msk, S)$.*
   (c) *Re-encryption oracle $\mathcal{O}_{re}(S, (M', \rho'), C_{(M,\rho)})$: on input an attribute set S, an access structure (M', ρ'), and an original ciphertext $C_{(M,\rho)}$, $\mathcal{C}$ returns $C^R_{(M',\rho')} \leftarrow ReEnc(rk_{S\rightarrow(M',\rho')}, C_{(M,\rho)})$ to $\mathcal{A}$, where $rk_{S\rightarrow(M',\rho')} \leftarrow ReKeyGen(sk_S, S, (M', \rho'))$, $sk_S \leftarrow KeyGen(msk, S)$ and $S \models (M, \rho)$.*
   (d) *Original ciphertext decryption oracle $\mathcal{O}_{d2}(S, C_{(M,\rho)})$: on input an attribute set S and an original ciphertext $C_{(M,\rho)}$, $\mathcal{C}$ returns $m \leftarrow Dec(S, sk_S, C_{(M,\rho)})$ to $\mathcal{A}$, where $sk_S \leftarrow KeyGen(msk, S)$ and $S \models (M, \rho)$.*
   (e) *Re-encrypted ciphertext decryption oracle $\mathcal{O}_{d1}(S', C^R_{(M',\rho')})$: on input an attribute set S' and a re-encrypted ciphertext $C^R_{(M',\rho')}$, $\mathcal{C}$ returns $m \leftarrow Dec_R(S', sk_{S'}, C^R_{(M',\rho')})$, where $sk_{S'} \leftarrow KeyGen(msk, S')$ and $S' \models (M', \rho')$.*

   Note that if the ciphertexts queried to oracles $\mathcal{O}_{re}$, $\mathcal{O}_{d2}$ and $\mathcal{O}_{d1}$ are invalid, $\mathcal{C}$ simply outputs $\perp$. In this phase the following queries are forbidden to issue:
   - *$\mathcal{O}_{sk}(S)$ for any $S \models (M^*, \rho^*)$; and*
   - *$\mathcal{O}_{rk}(S, (M', \rho'))$ for any $S \models (M^*, \rho^*)$, and $\mathcal{O}_{sk}(S')$ for any $S' \models (M', \rho')$.*
4. ***Challenge.*** *$\mathcal{A}$ outputs two equal length messages m_0 and m_1 to $\mathcal{C}$. $\mathcal{C}$ returns $C^*_{(M^*,\rho^*)} = Enc((M^*, \rho^*), m_b)$ to $\mathcal{A}$, where $b \in_R \{0, 1\}$.*
5. ***Query Phase II.*** *$\mathcal{A}$ continues making queries as in Query Phase I except the following:*
   (a) *$\mathcal{O}_{sk}(S)$ for any $S \models (M^*, \rho^*)$;*
   (b) *$\mathcal{O}_{rk}(S, (M', \rho'))$ for any $S \models (M^*, \rho^*)$, and $\mathcal{O}_{sk}(S')$ for any $S' \models (M', \rho')$;*
   (c) *$\mathcal{O}_{re}(S, (M', \rho'), C^*_{(M^*,\rho^*)})$ for any $S \models (M^*, \rho^*)$, and $\mathcal{O}_{sk}(S')$ for any $S' \models (M', \rho')$;*
   (d) *$\mathcal{O}_{d2}(S, C^*_{(M^*,\rho^*)})$ for any $S \models (M^*, \rho^*)$; and*
   (e) *$\mathcal{O}_{d1}(S', C^R_{(M',\rho')})$ for any $C^R_{(M',\rho')}$, $S' \models (M', \rho')$, where $C^R_{(M',\rho')}$ is a derivative of $C^*_{(M^*,\rho^*)}$. As of [8], the derivative of $C^*_{(M^*,\rho^*)}$ is defined as follows.*
      i. *$C^*_{(M^*,\rho^*)}$ is a derivative of itself.*
      ii. *If $\mathcal{A}$ has issued a re-encryption key query on $(S, (M', \rho'))$ to obtain the re-encryption key $rk_{S\rightarrow(M',\rho')}$, and achieved $C^R_{(M',\rho')} \leftarrow ReEnc(rk_{S\rightarrow(M',\rho')}, C^*_{(M^*,\rho^*)})$, then $C^R_{(M',\rho')}$ is a derivative of $C^*_{(M^*,\rho^*)}$, where $S \models (M^*, \rho^*)$.*
      iii. *If $\mathcal{A}$ has issued a re-encryption query on $(S, (M', \rho'), C^*_{(M^*,\rho^*)})$ and obtained the re-encrypted ciphertext $C^R_{(M',\rho')}$, then $C^R_{(M',\rho')}$ is a derivative of $C^*_{(M^*,\rho^*)}$, where $S \models (M^*, \rho^*)$.*
6. ***Guess.*** *$\mathcal{A}$ outputs a guess bit $b' \in \{0, 1\}$. If $b' = b$, $\mathcal{A}$ wins.*

The advantage of $\mathcal{A}$ is defined as $\epsilon_1 = Adv^{IND-sAS-CCA-Or}_{CP-ABPRE,\mathcal{A}}(1^k, \mathcal{U}) = |Pr[b' = b] - \frac{1}{2}|$.

Remarks. The model above can be converted to the IND-aAS-CCA-Or game by allowing $\mathcal{A}$ to output the challenge access structure (M^*, ρ^*) in the challenge phase. Meanwhile, there is no restriction for $\mathcal{A}$ in Query Phase I. Besides, $\mathcal{C}$ will output the challenge ciphertext if the forbidden queries defined in Query Phase I of the above game are never made.

The definition of IND-sAS-CCA-Re security can be defined in an orthogonal manner as follows.

Definition 5. *A single-hop unidirectional CP-ABPRE scheme is IND-sAS-CCA secure at re-encrypted ciphertext if the advantage $\epsilon_2 = Adv^{IND-sAS-CCA-Re}_{CP-ABPRE,\mathcal{A}}(1^k, \mathcal{U})$ is negligible for any PPT adversary $\mathcal{A}$*

in the following experiment. Set $O_1 = \{\mathcal{O}_{sk}, \mathcal{O}_{rk}, \mathcal{O}_{re}, \mathcal{O}_{d2}, \mathcal{O}_{d1}\}$.

$$\epsilon_2 = |Pr[b' = b : ((M^*, \rho^*), State_1) \leftarrow \mathcal{A}(1^k); (param, msk) \leftarrow Setup(1^k, \mathcal{U}); (m_0, m_1, (M, \rho), State_2) \leftarrow \mathcal{A}^{O_1}(param, State_1); b \in_R \{0,1\}; C^{R*}_{(M^*,\rho^*)} \leftarrow ReEnc(rk_{S\rightarrow(M^*,\rho^*)}, C_{(M,\rho)}); b' \leftarrow A^{O_1}(C^{R*}_{(M^*,\rho^*)}, State_2)] - \frac{1}{2}|,$$

where $State_1$ *and* $State_2$ *are the state information,* (M, ρ) *and* (M^*, ρ^*) *are disjoint,* (M^*, ρ^*) *is the challenge access structure,* $S \models (M, \rho)$, $rk_{S\rightarrow(M^*,\rho^*)} \leftarrow ReKeyGen(sk_S, S, (M^*, \rho^*))$, $C_{(M,\rho)} \leftarrow Enc((M, \rho), m_b)$, $\mathcal{O}_{sk}, \mathcal{O}_{rk}, \mathcal{O}_{re}, \mathcal{O}_{d2}, \mathcal{O}_{d1}$ *are the oracles defined in Definition 4. However, these oracles are restricted by the following constraints. For* $\mathcal{O}_{sk}$*, the query on* S *is forbidden to issue for any* $S \models (M^*, \rho^*)$*. For* $\mathcal{O}_{rk}$*, it works as in the IND-sAS-CCA-Or game.* $\mathcal{O}_{re}$ *will output* $\perp$ *if* $\mathcal{A}$ *queries invalid original ciphertexts or re-encrypted ciphertexts. There is no restriction for* $\mathcal{O}_{d2}$ *except that the oracle will reject invalid original ciphertexts. If* $\mathcal{A}$ *queries to* $\mathcal{O}_{d1}$ *on* $(S, C^{R*}_{(M^*,\rho^*)})$ *or invalid re-encrypted ciphertexts, the oracle outputs* $\perp$*, where* $S \models (M^*, \rho^*)$.

Remarks. In Definition 5 $\mathcal{O}_{rk}$ must follow the constraints defined in Definition 4. This is necessary because in selective access structure model the challenger cannot construct a valid private key for any $S \models (M^*, \rho^*)$. Thus the re-encryption key $rk_{S\rightarrow(M',\rho')}$ has to be randomly generated, where $\mathcal{A}$ is allowed to query $\mathcal{O}_{sk}(S')$ for any $S' \models (M', \rho')$. If such a re-encryption key can be issued by $\mathcal{A}$, then $\mathcal{A}$ can distinguish the simulation from the real attack[9]. In this case, to generate the corresponding re-encryption, $\mathcal{O}_{re}$ must be provided for $\mathcal{A}$. Note that Definition 5 can be regarded as a weaker notion when compared with the (adaptive) re-encrypted ciphertext security model defined in traditional PRE.

The model above can be also converted to the IND-aAS-CCA-Re game as follows. $\mathcal{A}$ is allowed to output (M^*, ρ^*) in the challenge phase. There is no restriction for $\mathcal{A}$ to query $\mathcal{O}_{rk}$. Besides, $\mathcal{O}_{re}$ is unnecessary as $\mathcal{A}$ is allowed to query any re-encryption key.

We now proceed to the *selective* collusion resistance for CP-ABPRE. Like collusion resistance defined in traditional PRE, this security notion also guarantees that a dishonest proxy cannot compromise the entire private key of the delegator even it colludes with the corresponding delegatee. However, an adversary is required to output an attribute set that it wishes to attack before the setup phase. The selective collusion resistance model can be defined via the identical manner introduced in [20,22], we hence omit the details. Instead, we prefer to show that the IND-sAS-CCA-Or security already implies selective collusion resistance.

Theorem 1. *Suppose a single-hop unidirectional CP-ABPRE scheme is IND-sAS-CCA-Or secure, then it is selective collusion resistant as well.*

Please refer to Appendix A for the proof of Theorem 1.

3 Preliminaries

We first give a brief review of bilinear maps and the decisional q-parallel BDHE assumption, and next introduce the target collision resistance hash function.

Bilinear Maps. Let *BSetup* denote an algorithm that, on input the security parameter 1^k, outputs the parameters for a bilinear map as $(p, g, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups with prime order $p \in \Theta(2^k)$ and g is a generator of $\mathbb{G}$. The efficient mapping $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ has three properties: (1) *Bilinearity*: for all $g \in \mathbb{G}$ and $a, b \in_R \mathbb{Z}^*_p$, $e(g^a, g^b) = e(g, g)^{ab}$; (2) *Non-degeneracy*: $e(g, g) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the unit of $\mathbb{G}_T$; (3) *Computability*: e can be efficiently computed.

[9] This is so because $\mathcal{A}$ can verify whether such a re-encryption key is valid or not as follows: $\mathcal{A}$ first generates a ciphertext of a chosen message m under (M^*, ρ^*), re-encrypts the ciphertext using the re-encryption key, and then decrypts the re-encrypted ciphertext using $sk_{S'}$. If the decryption outputs m, then the re-encryption key is valid.

Definition 6. ***Decisional q-parallel BDHE Assumption [26].*** *Given a tuple $y=$*

$$g, g^s, g^a, ..., g^{a^q}, g^{a^{q+2}}, ..., g^{a^{2q}}$$
$$\forall_{1\leq j\leq q}\ g^{s\cdot b_j}, g^{a/b_j}, ..., g^{a^q/b_j}, g^{a^{q+2}/b_j}, ..., g^{a^{2q}/b_j}$$
$$\forall_{1\leq j,k\leq q, k\neq j}\ g^{a\cdot s\cdot b_k/b_j}, ..., g^{a^q\cdot s\cdot b_k/b_j},$$

the decisional q-parallel BDHE problem is to decide whether $T = e(g,g)^{a^{q+1}\cdot s}$, where $a, s, b_1, ..., b_q \in_R \mathbb{Z}_p, T \in_R \mathbb{G}_T$ and g is a generator of $\mathbb{G}$. Define $Adv_{\mathcal{A}}^{D-q-parallelBDHE} = |Pr[\mathcal{A}(y, e(g,g)^{a^{q+1}\cdot s}) = 0] - Pr[\mathcal{A}(y,T) = 0]|$ as the advantage of adversary $\mathcal{A}$ in winning the decisional q-parallel BDHE problem. We say that the decisional q-parallel BDHE assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ if no PPT algorithm has non-negligible advantage.

Target Collision Resistant Hash Function. Target Collision Resistant (TCR) hash function was introduced by Cramer and Shoup [12]. A TCR hash function H guarantees that given a random element x which is from the valid domain of H, a PPT adversary A cannot find $y \neq x$ such that $H(x) = H(y)$. We let $Adv_{H,A}^{TCR} = Pr[(x,y) \leftarrow \mathcal{A}(1^k) : H(x) = H(y), x \neq y, x, y \in DH]$ be the advantage of A in successfully finding collisions from a TCR hash function H, where DH is the valid input domain of H, k is the security parameter. If a hash function is chosen from a TCR hash function family, $Adv_{H,A}^{TCR}$ is negligible.

4 A New CP-ABPRE with CCA Security

In this section we construct a new CP-ABPRE in the random oracle model with CCA security. Prior to proposing the scheme, we first introduce some intuition behind our construction. We choose Waters ABE (the most efficient construction proposed in [26]) as a basic building block of our scheme due to the following reasons. The construction of Waters ABE scheme enables us to convert the scheme to be an ABE Key Encapsulation in the random oracle model. Specifically, in our construction a content key that is asymmetrically encrypted under an access policy is used to hide a message in a symmetric way. Furthermore, Waters ABE scheme utilizes LSSS to support any monotonic access formula for ciphertexts. It is a desirable property for CP-ABPRE systems when being implemented in practice. In addition, the construction for ciphertexts, whose size is linear in the size of formula, is able to help us relieve the communication cost incurred by re-encryption and the generation of re-encryption key.

CCA Security. As discussed in Section 1.1, the biggest challenge would be how to achieve CCA security while not jeopardizing the properties of attribute-based re-encryption, unidirectionality and collusion resistance. In our construction, we use some technique, which is like the FO [14] conversion, to capture CCA security. Specifically, in the construction of ciphertext we utilize a TCR hash function to "sign" the ciphertext's components as well as the description of LSSS, and meanwhile, construct a "verification key" to check the validity of such a "signature". In algorithm *Enc*, it can be seen that the "signature" is D and the verification key is A_3 such that the validity of ciphertext can be checked by $e(A_2, g_1) \stackrel{?}{=} e(g, A_3)$ and $e(A_3, H_4(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho))) \stackrel{?}{=} e(g_1, D), S \stackrel{?}{\models} (M, \rho)$. In algorithm *ReEnc*, the proxy could first check the above equations so as to guarantee the re-encryption to intake correct values. Besides the well-formness of the components $\{(B_i, C_i) | 1 \leq i \leq l\}$ should be verified as they are the input for re-encryption as well. To capture such a requirement, we let the proxy check $e(\prod_{i\in I} B_i^{w_i}, g) \stackrel{?}{=} e(A_2, g^a) \cdot \prod_{i\in I}(e(C_i^{-1}, H_3(\rho(i))^{w_i}))$. After fulfilling the re-encryption, the proxy will output the re-encrypted ciphertext (S, (M,ρ), A_1, A_3, (B_1, C_1), ..., (B_l, C_l), D, A_4, rk_4). For a legitimate delegatee, he/she is able to check the validity of the re-encrypted ciphertext as (M,ρ), A_1, A_3, (B_1, C_1), ..., (B_l, C_l) are "signed" by D and S is "signed" in rk_4 (which will be further elaborated later). Besides, A_4 is tightly related to the original ciphertext's components A_1 and A_3 due to a fact that $A_3 \stackrel{?}{=} g_1^{H_1(m,\beta)}$ can tell whether A_4 is mutated or not.

As to the generation of re-encryption key $rk_{S\to(M',\rho')}$ from an attribute set S to a new access policy (M',ρ'), it can be seen that S and (M',ρ') are "signed" in D', and the signature can be checked by A'_2. rk_1, rk_3, R_x are tightly related to rk_4 via δ, and rk_1 is bound with rk_2 with θ, where rk_4 is the encryption of δ under (M',ρ') with CCA security. Here if rk_1, rk_2, rk_3 and R_x are mutated by an adversary, the corresponding re-encryption will yield an invalid results corresponding to rk_4; on the other hand, if the description of S and (M',ρ'), and the encryption rk_4 are mutated, the proxy can tell by checking $e(A'_2, H_6(A'_1, A'_2, (B'_1, C'_1), ..., (B'_{l'}, C'_{l'}), S, (M',\rho'))) \stackrel{?}{=} e(g, D')$. Thus the construction precludes an adversary from generating a new and valid re-encryption key $rk_{S'\to(M',\rho')}$ or $rk_{S\to(M'',\rho'')}$ based on knowledge of $rk_{S\to(M',\rho')}$.

Taking a more specific look at the re-encryption key, we can see that only the private key of the delegator is taken as an input. Accordingly, our scheme is non-interactive in the generation of re-encryption key (which saves the bandwidth of communication) and unidirectional. Due to the construction of rk_1 an adversary cannot compromise the entire private key of the delegator without knowledge of θ even if it colludes with the corresponding delegatee. This captures collusion resistance. As to single-hop property, it can be achieved as follows. Algorithm *ReEnc* shows that A_3 (i.e. g_1^s) is a necessary component for re-encryption. However, such a component is not included in rk_4 such that rk_4 is out of the capability of being re-encrypted. Thus our scheme is single-hop as well.

Note that the definition of the relevant variables used above can be seen in our scheme.

The description of our new CP-ABPRE scheme with CCA security is as follows. Unless stated otherwise, we let $\mathcal{U}$ be the attributes universe in the system, and S be an attribute set, $S \subseteq \mathcal{U}$.

1. $Setup(1^k, \mathcal{U})$. Given a security parameter k and $\mathcal{U}$, run $(p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow BSetup(1^k)$. Choose two random values $a, \alpha \in \mathbb{Z}_p^*$, a random generator $g_1 \in \mathbb{G}$, and set the following TCR hash functions $H_1 : \{0,1\}^{2k} \to \mathbb{Z}_p^*$, $H_2 : \mathbb{G}_T \to \{0,1\}^{2k}$, $H_3 : \{0,1\}^* \to \mathbb{G}$, $H_4 : \{0,1\}^* \to \mathbb{G}$, $H_5 : \{0,1\}^k \to \mathbb{Z}_p^*$, $H_6 : \{0,1\}^* \to \mathbb{G}$. The public parameters are $param = (p,\ g,\ \mathbb{G},\ \mathbb{G}_T,\ e,\ g_1,\ g^a,\ e(g,g)^\alpha,\ H_1,\ H_2,\ H_3,\ H_4,\ H_5,\ H_6)$, and the master secret key is $msk = g^\alpha$.
2. $KeyGen(msk, S)$. Given msk and an attribute set S, choose $t \in_R \mathbb{Z}_p^*$, and set the private key sk_S as
$$K = g^{a\cdot t} \cdot g^\alpha, L = g^t, \forall x \in S\ K_x = H_3(x)^t.$$
3. $Enc((M,\rho), m)$. Taking an LSSS access structure (M,ρ) (M is an $l \times n$ matrix, and the function ρ associates rows of M to attributes) and a message $m \in \{0,1\}^k$ as input, the encryption algorithm works as follows.
   (a) Choose $\beta \in_R \{0,1\}^k$, set $s = H_1(m, \beta)$ and a random vector $v = (s, y_2, ..., y_n)$, where $y_2, ..., y_n \in_R \mathbb{Z}_p^*$.
   (b) For $i = 1$ to l, set $\lambda_i = v \cdot M_i$, where M_i is the vector corresponding to the ith row of M.
   (c) Choose $r_1, ..., r_l \in_R \mathbb{Z}_p^*$, set
$$A_1 = (m||\beta) \oplus H_2(e(g,g)^{\alpha\cdot s}), A_2 = g^s, A_3 = g_1^s, B_1 = (g^a)^{\lambda_1} \cdot H_3(\rho(1))^{-r_1}, C_1 = g^{r_1}, ...,$$
$$B_l = (g^a)^{\lambda_l} \cdot H_3(\rho(l))^{-r_l}, C_l = g^{r_l}, D = H_4(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M,\rho))^s,$$
   and output the original ciphertext $C_{(M,\rho)} = ((M,\rho),\ A_1,\ A_2,\ A_3,\ (B_1, C_1),\ ...,\ (B_l, C_l),\ D)$. Note that $\{\rho(i) | 1 \le i \le l\}$ are the attributes used in the access structure (M,ρ). Like [26], we allow an attribute to be associated with multiple rows of matrix M, i.e. the function ρ is not injective.
4. $ReKeyGen(sk_S, S, (M',\rho'))$. Given a private key $sk_S = (K, L, K_x)$ and the corresponding attribute set S, and an LSSS access structure (M',ρ'), the re-encryption key is generated as follows, where $x \in S$, M' is an $l' \times n'$ matrix, and the function ρ' associates rows of M' to attributes.
   – The delegator does the following encryption:
   (a) Choose $\beta', \delta \in_R \{0,1\}^k$, set $s' = H_1(\delta, \beta')$ and a random vector $v' = (s', y'_2, ..., y'_{n'})$, where $y'_2, ..., y'_{n'} \in_R \mathbb{Z}_p^*$.

(b) For $i = 1$ to l', set $\lambda'_i = v' \cdot M'_i$, where M'_i is the vector corresponding to the ith row of M'.
(c) Choose r'_1, ..., $r'_{l'} \in_R \mathbb{Z}_p^*$, compute $A'_1 = (\delta||\beta') \oplus H_2(e(g,g)^{\alpha \cdot s'})$, $A'_2 = g^{s'}$, $B'_1 = (g^a)^{\lambda'_1} \cdot H_3(\rho'(1))^{-r'_1}$, $C'_1 = g^{r'_1}$, ..., $B'_{l'} = (g^a)^{\lambda'_{l'}} \cdot H_3(\rho'(l'))^{-r'_{l'}}$, $C'_{l'} = g^{r'_{l'}}$, $D' = H_6(A'_1, A'_2, (B'_1, C'_1), ..., (B'_{l'}, C'_{l'}), S, (M', \rho'))^{s'}$, and output the ciphertext $C_{(M',\rho')} = ((M', \rho'), A'_1, A'_2, (B'_1, C'_1), ..., (B'_{l'}, C'_{l'}), D')$.

– The delegator chooses $\theta \in_R \mathbb{Z}_p^*$, and sets $rk_1 = K^{H_5(\delta)} \cdot g_1^{\theta}, rk_2 = g^{\theta}, rk_3 = L^{H_5(\delta)}, \forall x \in S\ R_x = K_x^{H_5(\delta)}, rk_4 = C_{(M',\rho')}$, and outputs the re-encryption key $rk_{S \to (M',\rho')} = (S, rk_1, rk_2, rk_3, rk_4, R_x)$.

5. $ReEnc(rk_{S \to (M',\rho')}, C_{(M,\rho)})$. Parse the original ciphertext $C_{(M,\rho)}$ as $((M, \rho), A_1, A_2, A_3, (B_1, C_1), ..., (B_l, C_l), D)$, and the re-encryption key $rk_{S \to (M',\rho')}$ as $(S, rk_1, rk_2, rk_3, rk_4, R_x)$. Recall that M is an $l \times n$ matrix. Let $I \subset \{1, ..., l\}$ be defined as $I = \{i : \rho(i) \in S\}$, $\{w_i \in \mathbb{Z}_p^*\}_{i \in I}$ be a set of constants such that $\sum_{i \in I} w_i \cdot \lambda_i = s$ if $\{\lambda_i\}$ are valid shares of any secret s according to M and $S \models (M, \rho)$[10].
(a) The proxy is able to verify whether the re-encryption key $rk_{S \to (M',\rho')}$ contains valid S and (M', ρ') or not by checking

$$e(A'_2, H_6(A'_1, A'_2, (B'_1, C'_1), ..., (B'_{l'}, C'_{l'}), S, (M', \rho'))) \stackrel{?}{=} e(g, D').$$

(b) Verify the validity of the original ciphertext

$$\begin{aligned} &e(A_2, g_1) \stackrel{?}{=} e(g, A_3), e(A_3, H_4(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho))) \stackrel{?}{=} e(g_1, D), S \stackrel{?}{\models} (M, \rho), \\ &e(\prod_{i \in I} B_i^{w_i}, g) \stackrel{?}{=} e(A_2, g^a) \cdot \prod_{i \in I} (e(C_i^{-1}, H_3(\rho(i))^{w_i})). \end{aligned} \tag{1}$$

If Eq. (1) does not hold, output $\perp$. Otherwise, proceed.
(c) Compute $A_4 = \frac{e(A_2, rk_1)/e(A_3, rk_2)}{(\prod_{i \in I}(e(B_i, rk_3) \cdot e(C_i, R_{\rho(i)}))^{w_i})}$, and output the re-encrypted ciphertext $C^R_{(M',\rho')} = (S, (M, \rho), A_1, A_3, (B_1, C_1), ..., (B_l, C_l), D, A_4, rk_4)$.

6. $Dec(S, sk_S, C_{(M,\rho)})$. Parse the original ciphertext $C_{(M,\rho)}$ as $((M, \rho), A_1, A_2, A_3, (B_1, C_1), ..., (B_l, C_l), D)$, and the private key sk_S (for an attribute set S) as (K, L, K_x) $(x \in S)$. Note that let $I \subset \{1, ..., l\}$ be defined as $I = \{i : \rho(i) \in S\}$, $\{w_i \in \mathbb{Z}_p^*\}_{i \in I}$ be a set of constants such that $\sum_{i \in I} w_i \cdot \lambda_i = s$.
(1) Verify Eq. (1). If Eq. (1) does not hold, output $\perp$. Otherwise, proceed.
(2) Compute $Z = e(A_2, K)/(\prod_{i \in I}(e(B_i, L) \cdot e(C_i, K_{\rho(i)}))^{w_i})$ and $m||\beta = H_2(Z) \oplus A_1$, output m if $A_3 = g_1^{H_1(m,\beta)}$, and output $\perp$ otherwise.

7. $Dec_R(S', sk_{S'}, C^R_{(M',\rho')})$. Parse the re-encrypted ciphertext $C^R_{(M',\rho')}$ as $(S, (M, \rho), A_1, A_3, (B_1, C_1), ..., (B_l, C_l), D, A_4, rk_4)$, and the private key $sk_{S'}$ (for an attribute set S') as (K', L', K'_x) $(x \in S')$.
(a) Recover $\delta||\beta'$ as follows. Let $I' \subset \{1, ..., l'\}$ be defined as $I' = \{i : \rho'(i) \in S'\}$, $\{w'_i \in \mathbb{Z}_p^*\}_{i \in I'}$ be a set of constants such that $\sum_{i \in I'} w'_i \cdot \lambda'_i = s'$ if $\{\lambda'_i\}$ are valid shares of any secret s' according to M' and $S' \models (M', \rho')$.
i. Verify

$$e(A'_2, H_6(A'_1, A'_2, (B'_1, C'_1), ..., (B'_{l'}, C'_{l'}), S, (M', \rho'))) \stackrel{?}{=} e(g, D'), S' \stackrel{?}{\models} (M', \rho'). \tag{2}$$

If Eq. (2) does not hold, output $\perp$. Otherwise, proceed.
ii. Compute $Z' = e(A'_2, K')/(\prod_{i \in I'}(e(B'_i, L') \cdot e(C'_i, K'_{\rho'(i)}))^{w'_i})$ and $\delta||\beta' = H_2(Z') \oplus A'_1$, proceed if $A'_2 = g^{H_1(\delta,\beta')}$, and output $\perp$ otherwise.

[10] As stated in [4,26], with knowledge of M and I one can find the values w_i satisfying $\sum_{i \in I} w_i \cdot \lambda_i = s$.

(b) Compute $m||\beta = H_2(A_4^{\frac{1}{H_5(\delta)}}) \oplus A_1$, output m if $A_3 = g_1^{H_1(m,\beta)}$, $D = H_4(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho))^{H_1(m,\beta)}$ and $S \models (M, \rho)$, and output $\perp$ otherwise.

– **Correctness for Original Ciphertext.**

$$\begin{aligned} Z &= e(A_2, K)/(\prod_{i\in I}(e(B_i, L) \cdot e(C_i, K_{\rho(i)}))^{w_i}) \\ &= \frac{e(g^s, g^{a\cdot t} \cdot g^{\alpha})}{(\prod_{i\in I}(e(g^{a\cdot\lambda_i} \cdot H_3(\rho(i))^{-r_i}, g^t) \cdot e(g^{r_i}, H_3(\rho(i))^t))^{w_i})} \\ &= \frac{e(g^s, g^{a\cdot t} \cdot g^{\alpha})}{e(g, g^{a\cdot t})^{\sum_{i\in I}\lambda_i\cdot w_i}} = e(g^s, g^{\alpha}), \end{aligned}$$

Hence, we have $H_2(Z) \oplus A_1 = H_2(e(g^s, g^{\alpha})) \oplus (m||\beta) \oplus H_2(e(g, g)^{\alpha\cdot s}) = m||\beta$.

– **Correctness for Re-Encrypted Ciphertext.**

$$\begin{aligned} A_4 &= \frac{e(A_2, rk_1)/e(A_3, rk_2)}{(\prod_{i\in I}(e(B_i, rk_3) \cdot e(C_i, R_{\rho(i)}))^{w_i})} \\ &= \frac{e(g^s, (g^{a\cdot t} \cdot g^{\alpha})^{H_5(\delta)} \cdot g_1^{\theta})/e(g_1^s, g^{\theta})}{\prod_{i\in I}(e((g^a)^{\lambda_i} \cdot H_3(\rho(i))^{-r_i}, (g^t)^{H_5(\delta)}) \cdot e(g^{r_i}, H_3(\rho(i))^{t\cdot H_5(\delta)}))^{w_i}} \\ &= \frac{e(g^s, g^{\alpha\cdot H_5(\delta)}) \cdot e(g^s, g^{a\cdot t\cdot H_5(\delta)})}{e(g, g^{a\cdot t\cdot H_5(\delta)})^{\sum_{i\in I}\lambda_i\cdot w_i}} = e(g^s, g^{\alpha\cdot H_5(\delta)}), \end{aligned}$$

Thus we have $H_2(A_4^{\frac{1}{H_5(\delta)}}) \oplus A_1 = H_2(e(g, g)^{\alpha\cdot s\cdot H_5(\delta)})^{\frac{1}{H_5(\delta)}} \oplus (m||\beta) \oplus H_2(e(g, g)^{\alpha\cdot s}) = m||\beta$.

Before giving the formal security analysis, we first give some intuition as to why the scheme above is secure against CCA. For the security of original ciphertext, let $C^*_{(M^*,\rho^*)} = ((M^*, \rho^*), A_1^*, A_2^*, A_3^*, (B_1^*, C_1^*), ..., (B_l^*, C_l^*), D^*)$ be the challenge ciphertext of m_b. Suppose an adversary $\mathcal{A}$ who follows the constraints defined in Definition 4 will try to get extra advantage in guessing the value of the bit b by using $\mathcal{O}_{re}$ and $\mathcal{O}_{d2}$. Specifically, $\mathcal{A}$ might mutate the challenge ciphertext, and submit the resulting ciphertext to $\mathcal{O}_{re}$ and $\mathcal{O}_{d2}$. From Eq. (1), such a change is noticeable with non-negligible probability. This is so because A_1^*, A_3^*, (B_1^*, C_1^*), ..., (B_l^*, C_l^*) are bound by D as well as the description of (M^*, ρ^*). Note that D can be viewed as a signature for such components. Besides, the integrity of A_2^* is bound by A_3^*. If the ciphertext is mutated, Eq. (1) will not hold. Therefore, no extra advantage in guessing b leaks to $\mathcal{A}$.

For the security of re-encrypted ciphertext, let $C^{R*}_{(M^*,\rho^*)} = (S, (M, \rho), A_1^*, A_3^*, (B_1^*, C_1^*), ..., (B_l^*, C_l^*), D^*, A_4^*, rk_4^*)$ be the re-encrypted ciphertext of m_b. Following Definition 5, $\mathcal{A}$ will try to gain extra advantage in winning the game with the help of $\mathcal{O}_{d1}$. Before proceeding, we show that the re-encrypted ciphertext cannot be re-encrypted, i.e. given $\mathcal{O}_{re}$ $\mathcal{A}$ cannot achieve extra advantage. Clearly, $rk_4^* = C_{(M^*,\rho^*)}$ cannot be re-encrypted without A_3' (i.e. $g_1^{s'}$) which is a necessary component for re-encryption. Furthermore, A_2^* that is needed in re-encryption and the verification in Eq. (1) is excluded in the re-encrypted ciphertext. Accordingly, the re-encryption query for the re-encrypted ciphertext will be rejected.

Given $C^{R*}_{(M^*,\rho^*)}$ $\mathcal{A}$ cannot mutate the ciphertext and issue the resulting ciphertext to $\mathcal{O}_{d1}$ such that the oracle outputs a valid decryption value without any rejection. The reason is that A_1^*, A_3^*, (B_1^*, C_1^*), ..., (B_l^*, C_l^*) are still bound by D^* as well as the description of (M, ρ); meanwhile, S and the description of (M^*, ρ^*) are bound by rk_4^*. Note that rk_4^* is secure against CCA[11]. Here the only

[11] It is not difficult to see that D' can be regarded as a signature for all the components contained in rk_4^* (except D' itself) and S, and A_2' can be seen as the verification key.

consideration left is the integrity of A_4^*. We state that if A_4^* is mutated by $\mathcal{A}$, the challenger can tell the change with non-negligible probability. Please refer to the proof for details. Hence $\mathcal{A}$ cannot acquire extra advantage by using $\mathcal{O}_{d1}$.

Therefore we have the following theorems.

Theorem 2. *Suppose the decisional q-parallel BDHE assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$*, and* H_1*,* H_2*,* H_3*,* H_4*,* H_5*,* H_6 *are the TCR hash functions, our CP-ABPRE scheme is IND-sAS-CCA-Or secure in the random oracle model.*

Please refer to Appendix B.1 for the proof of Theorem 2.

Theorem 3. *Suppose the decisional q-parallel BDHE assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$*, and* H_1*,* H_2*,* H_3*,* H_4*,* H_5*,* H_6 *are the TCR hash functions, our CP-ABPRE scheme is IND-sAS-CCA-Re secure in the random oracle model.*

Please refer to Appendix B.2 for the proof of Theorem 3.

5 Concluding Remarks

In this paper, we proposed a new single-hop unidirectional CP-ABPRE scheme, which supports attribute-based re-encryption with any monotonic access structure, to tackle the open problem left by the existing CP-ABPRE schemes. We also showed that our scheme can be proved IND-sAS-CCA secure in the random oracle model assuming the decisional q-parallel BDHE assumption holds.

Removing the ROM. The technique introduced in [19,27] might be a possible approach to remove random oracles. We leave this as our future work.

This paper also motivates some interesting open problems, for example, how to construct a CCA secure CP-ABPRE scheme in the adaptive access structure model, i.e. achieving IND-aAS-CCA security.

References

1. Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. In *NDSS*. The Internet Society, 2005.
2. Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.*, 9(1):1–30, 2006.
3. Nuttapong Attrapadung, Javier Herranz, Fabien Laguillaumie, Benoît Libert, Elie de Panafieu, and Carla Rafols. Attribute-based encryption schemes with constant-size ciphertexts. *Theoretical Computer Science*, 422(0):15–38, 2012.
4. Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution.* PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.
5. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society, 2007.
6. Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In Kaisa Nyberg, editor, *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 127–144. Springer, 1998.
7. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004.
8. Ran Canetti and Susan Hohenberger. Chosen-ciphertext secure proxy re-encryption. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM Conference on Computer and Communications Security*, pages 185–194. ACM, 2007.
9. Ran Canetti, Hugo Krawczyk, and Jesper Buus Nielsen. Relaxing chosen-ciphertext security. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582. Springer, 2003.
10. Ling Cheung and Calvin C. Newport. Provably secure ciphertext policy ABE. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM Conference on Computer and Communications Security*, pages 456–465. ACM, 2007.

11. Cheng-Kang Chu and Wen-Guey Tzeng. Identity-based proxy re-encryption without random oracles. In Juan A. Garay, Arjen K. Lenstra, Masahiro Mambo, and René Peralta, editors, *ISC*, volume 4779 of *Lecture Notes in Computer Science*, pages 189–202. Springer, 2007.
12. Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.*, 33(1):167–226, January 2004.
13. Keita Emura, Atsuko Miyaji, and Kazumasa Omote. A timed-release proxy re-encryption scheme. *IEICE Transactions*, 94-A(8):1682–1695, 2011.
14. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *J. Cryptology*, 26(1):80–101, 2013.
15. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.
16. Goichiro Hanaoka, Yutaka Kawai, Noboru Kunihiro, Takahiro Matsuda, Jian Weng, Rui Zhang, and Yunlei Zhao. Generic construction of chosen ciphertext secure proxy re-encryption. In Orr Dunkelman, editor, *Topics in Cryptology - CT-RSA 2012*, volume 7178 of *Lecture Notes in Computer Science*, pages 349–364. Springer Berlin Heidelberg, 2012.
17. Anca-Andreea Ivan and Yevgeniy Dodis. Proxy cryptography revisited. In *NDSS*. The Internet Society, 2003.
18. Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2010.
19. Kaitai Liang, Zhen Liu, Xiao Tan, Duncan S. Wong, and Chunming Tang. A cca-secure identity-based conditional proxy re-encryption without random oracles. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *ICISC*, volume 7839 of *Lecture Notes in Computer Science*, pages 231–246. Springer, 2012.
20. Xiaohui Liang, Zhenfu Cao, Huang Lin, and Jun Shao. Attribute based proxy re-encryption with delegating capabilities. In Wanqing Li, Willy Susilo, Udaya Kiran Tupakula, Reihaneh Safavi-Naini, and Vijay Varadharajan, editors, *ASIACCS*, pages 276–286. ACM, 2009.
21. Benoît Libert and Damien Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. In Ronald Cramer, editor, *Public Key Cryptography*, volume 4939 of *Lecture Notes in Computer Science*, pages 360–379. Springer, 2008.
22. Song Luo, Jian bin Hu, and Zhong Chen. Ciphertext policy attribute-based proxy re-encryption. In Miguel Soriano, Sihan Qing, and Javier López, editors, *ICICS*, volume 6476 of *Lecture Notes in Computer Science*, pages 401–415. Springer, 2010.
23. Masahiro Mambo and Eiji Okamoto. Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. *IEICE Transactions*, E80-A(1):54–63, 1997.
24. Takeo Mizuno and Hiroshi Doi. Hybrid proxy re-encryption scheme for attribute-based encryption. In Feng Bao, Moti Yung, Dongdai Lin, and Jiwu Jing, editors, *Information Security and Cryptology*, volume 6151 of *Lecture Notes in Computer Science*, pages 288–302. Springer Berlin Heidelberg, 2011.
25. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer Berlin Heidelberg, 2005.
26. Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2011.
27. Jian Weng, Minrong Chen, Yanjiang Yang, Robert H. Deng, Kefei Chen, and Feng Bao. CCA-secure unidirectional proxy re-encryption in the adaptive corruption model without random oracles. *Science China Information Sciences*, 53(3):593–606, 2010.

A Proof of Theorem 1

Proof. In the IND-sAS-CCA-Or game, $\mathcal{A}$ can achieve the following re-encryption keys from $\mathcal{O}_{rk}$: $rk_{S\to(M',\rho')}$ and $rk_{S'\to(M'',\rho'')}$, where $S \models (M^*,\rho^*)$ and $S' \models (M',\rho')$. Following the restrictions defined in the game, $\mathcal{A}$ cannot query the private key $sk_{S'}$ for any $S' \models (M',\rho')$[12] but the private key $sk_{S''}$ for any $S'' \models (M'',\rho'')$.

Suppose an IND-sAS-CCA-Or secure CP-ABPRE scheme is not selective collusion resistant. Then $\mathcal{A}$ is able to compromise the private key $sk_{S'}$ from $rk_{S'\to(M'',\rho'')}$ and $sk_{S''}$. Using $rk_{S\to(M',\rho')}$, $\mathcal{A}$ can re-encrypt the challenge ciphertext $C^*_{(M^*,\rho^*)}$ to $C^R_{(M',\rho')}$. $\mathcal{A}$ then decrypts the re-encrypted ciphertext with $sk_{S'}$ so as to output the value of the bit b. This contradicts the IND-sAS-CCA-Or security.

This completes the proof of Theorem 1. □

[12] $\mathcal{A}$ is forbidden to query any private key sk_S for any $S \models (M^*,\rho^*)$ as well.

B Security Analysis of CP-ABPRE

B.1 Proof of Theorem 2

Proof. Suppose there exists an adversary $\mathcal{A}$ who can break the IND-sAS-CCA-Or security of our scheme. We then construct a reduction algorithm $\mathcal{C}$ to decide whether $T = e(g,g)^{a^{q+1}\cdot s}$ or $T \in_R \mathbb{G}_T$. $\mathcal{C}$ plays the IND-sAS-CCA-Or game with $\mathcal{A}$ as follows.

$\mathcal{C}$ takes in $(p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow BSetup(1^k)$ and a q-parallel BDHE instance y and $T \in \mathbb{G}_T$, where T is either equal to $e(g,g)^{a^{q+1}\cdot s}$ or to $T' \in_R \mathbb{G}_T$.

1. **Initialization.** $\mathcal{A}$ outputs the challenge access structure (M^*, ρ^*) to $\mathcal{C}$, where M^* is an $l^* \times n^*$ matrix, $l^*, n^* \leq q$.
2. **Setup.** $\mathcal{C}$ chooses $\alpha', \gamma \in_R \mathbb{Z}_p^*$ and sets $g_1 = g^\gamma$, $e(g,g)^\alpha = e(g^a, g^{a^q}) \cdot e(g, g^{\alpha'})$[13]. Then $\mathcal{C}$ chooses the TCR hash functions as in the real scheme, and sends the public parameters $param = (p, g, \mathbb{G}, \mathbb{G}_T, e, g_1, g^a, e(g,g)^\alpha, H_1, H_2, H_3, H_4, H_5, H_6)$ to $\mathcal{A}$. From the point of view of $\mathcal{A}$, the public parameters are identical to those of the real scheme. At any time, $\mathcal{A}$ can adaptively query the random oracles H_j ($j \in \{1, ..., 6\}$) which are controlled by $\mathcal{C}$. $\mathcal{C}$ maintains the lists H_j^{List} ($j \in \{1, ..., 6\}$) which are initially empty and answers the queries to the random oracles as follows.
   (a) H_1: on receipt of an H_1 query on (m, β), if there is a tuple (m, β, s) in H_1^{List}, $\mathcal{C}$ forwards the predefined value s to $\mathcal{A}$, where $s \in \mathbb{Z}_p^*$. Otherwise, $\mathcal{C}$ sets $H_1(m, \beta) = s$, responds s to $\mathcal{A}$ and adds the tuple (m, β, s) to H_1^{List}, where $s \in_R \mathbb{Z}_p^*$.
   (b) H_2: on receipt of an H_2 query on $R \in \mathbb{G}_T$, if there is a tuple (R, δ_1) in H_2^{List}, $\mathcal{C}$ forwards the predefined value δ_1 to $\mathcal{A}$, where $\delta_1 \in \{0,1\}^{2k}$. Otherwise, $\mathcal{C}$ sets $H_2(R) = \delta_1$, responds δ_1 to $\mathcal{A}$ and adds the tuple (R, δ_1) to H_2^{List}, where $\delta_1 \in_R \{0,1\}^{2k}$.
   (c) H_3: on receipt of an H_3 query on $x \in \mathcal{U}$, if there is a tuple $(x, z_x, \delta_{2,x})$ in H_3^{List}, $\mathcal{C}$ forwards the predefined value $\delta_{2,x}$ to $\mathcal{A}$, where $z_x \in \mathbb{Z}_p^*$, $\delta_{2,x} \in \mathbb{G}$. Otherwise, $\mathcal{C}$ constructs $\delta_{2,x}$ as follows. Let X denote the set of indices i such that $\rho^*(i) = x$, where $1 \leq i \leq l^*$. Namely, X contains the indices of rows of matrix M^* that corresponds to the same attribute x. $\mathcal{C}$ chooses $z_x \in_R \mathbb{Z}_p^*$ and sets
   $$\delta_{2,x} = g^{z_x} \cdot \prod_{i \in X} g^{a \cdot M_{i,1}^*/b_i + a^2 \cdot M_{i,2}^*/b_i + \cdots + a^{n^*} \cdot M_{i,n^*}^*/b_i}.$$
   If $X = \emptyset$, $\mathcal{C}$ sets $\delta_{2,x} = g^{z_x}$. $\mathcal{C}$ responds $\delta_{2,x}$ to $\mathcal{A}$ and adds the tuple $(x, z_x, \delta_{2,x})$ to H_3^{List}.
   (d) H_4: on receipt of an H_4 query on $(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho))$, if there is a tuple $(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho), \xi_1, \delta_3)$ in H_4^{List}, $\mathcal{C}$ forwards the predefined value δ_3 to $\mathcal{A}$, where $\xi_1 \in \mathbb{Z}_p^*$, $\delta_3 \in \mathbb{G}$. Otherwise, $\mathcal{C}$ sets $\delta_3 = g^{\xi_1}$, responds δ_3 to $\mathcal{A}$ and adds the tuple $(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho), \xi_1, \delta_3)$ to H_4^{List}, where $\xi_1 \in_R \mathbb{Z}_p^*$.
   (e) H_5: on receipt of an H_5 query on $\delta \in \{0,1\}^k$, if there is a tuple (δ, ξ_2) in H_5^{List}, $\mathcal{C}$ forwards the predefined value ξ_2 to $\mathcal{A}$, where $\xi_2 \in \mathbb{Z}_p^*$. Otherwise, $\mathcal{C}$ sets $H_5(\delta) = \xi_2$, responds ξ_2 to $\mathcal{A}$ and adds the tuple (δ, ξ_2) to H_5^{List}, where $\xi_2 \in_R \mathbb{Z}_p^*$.
   (f) H_6: on receipt of an H_6 query on $(A_1', A_2', (B_1', C_1'), ..., (B_{l'}', C_{l'}'), S, (M', \rho'))$, if there is a tuple $(A_1', A_2', (B_1', C_1'), ..., (B_{l'}', C_{l'}'), S, (M', \rho'), \xi_3, \delta_4)$ in H_6^{List}, $\mathcal{C}$ forwards the predefined value δ_4 to $\mathcal{A}$, where $\xi_3 \in \mathbb{Z}_p^*$, $\delta_4 \in \mathbb{G}$. Otherwise, $\mathcal{C}$ sets $\delta_4 = g^{\xi_3}$, responds δ_4 to $\mathcal{A}$ and adds the tuple $(A_1', A_2', (B_1', C_1'), ..., (B_{l'}', C_{l'}'), S, (M', \rho'), \xi_3, \delta_4)$ to H_6^{List}, where $\xi_3 \in_R \mathbb{Z}_p^*$.
   In addition, $\mathcal{C}$ also maintains the following lists which are initially empty.
   (a) SK^{List}: records the tuples (S, sk_S), which are the results of the queries to $\mathcal{O}_{sk}(S)$.
   (b) RK^{List}: records the tuples $(S, (M', \rho'), \delta, \beta', rk_{S \to (M', \rho')}, tag_1, tag_2, tag_3)$, which are the results of the queries to $\mathcal{O}_{rk}(S, (M', \rho'))$, where tag_1, tag_2, tag_3 denote that the re-encryption key is randomly chosen, generated in $\mathcal{O}_{re}$ or in $\mathcal{O}_{rk}$, respectively.

[13] It can be seen that $\alpha = \alpha' + a^{q+1}$ (which cannot be computed by $\mathcal{C}$).

(c) RE^{List}: records the tuples $(S, (M', \rho'), C^R_{(M',\rho')}, tag_1, tag_2, tag_3)$, which are the results of the queries to $\mathcal{O}_{re}(S, (M', \rho'), C_{(M,\rho)})$, where tag_1, tag_2, tag_3 denote that the re-encrypted ciphertext is generated under a valid re-encryption key, under a randomly chosen re-encryption key or generated without any re-encryption key.

3. **Query Phase I.** $\mathcal{A}$ issues a series of queries to which $\mathcal{C}$ responds as follows.

(a) *Private key extraction oracle* $\mathcal{O}_{sk}(S)$: $\mathcal{C}$ constructs the private key sk_S for an attribute set S as follows. If $S \models (M^*, \rho^*)$, then $\mathcal{C}$ outputs a random bit in $\{0, 1\}$ and aborts the simulation (due to the restrictions defined in Definition 4). Otherwise, that is $S \not\models (M^*, \rho^*)$, $\mathcal{C}$ chooses $r \in_R \mathbb{Z}_p^*$, $w = (w_1, ..., w_{n^*}) \in \mathbb{Z}_p^{*n^*}$ such that $w_1 = -1$ and $\forall i, \rho^*(i) \in S$ we have that $w \cdot M_i^* = 0$[14]. $\mathcal{C}$ then sets $L = g^r \cdot \prod_{i=1,...,n^*} g^{a^{q+1-i} \cdot w_i} = g^t$. Here t is implicitly defined as $t = r + w_1 \cdot a^q + \cdots + w_{n^*} \cdot a^{q-n^*+1}$. $\mathcal{C}$ further constructs K as $K = g^{\alpha'} \cdot g^{a \cdot r} \cdot \prod_{i=2,...,n^*} g^{a^{q+2-i} \cdot w_i}$. One can verify that K is valid

$$\begin{aligned} K &= g^{\alpha'} \cdot g^{a \cdot r} \cdot \prod_{i=2,...,n^*} g^{a^{q+2-i} \cdot w_i} = g^{\alpha'} \cdot g^{a^{q+1}} \cdot g^{-a^{q+1}} \cdot g^{a \cdot r} \cdot \prod_{i=2,...,n^*} g^{a^{q+2-i} \cdot w_i} \\ &= g^{\alpha} \cdot (g^r \cdot \prod_{i=1,...,n^*} g^{a^{q+1-i} \cdot w_i})^a = g^{\alpha} \cdot L^a = g^{\alpha} \cdot g^{a \cdot t}. \end{aligned}$$

If $x \in S$ but $\rho^*(i) \neq x$ for any $i \in \{1, ..., l^*\}$, then $\mathcal{C}$ sets $K_x = L^{z_x}$. It is easily to see that $K_x = L^{z_x} = (g^t)^{z_x} = \delta_{2,x}^t = H_3(x)^t$. Otherwise, $\mathcal{C}$ constructs K_x as $K_x = L^{z_x} \cdot \prod_{i \in X} \prod_{j=1,...,n^*} (g^{(a^j/b_i) \cdot r} \cdot \prod_{k=1,...,n^*, k \neq j} (g^{a^{q+1+j-k}/b_i})^{w_k})^{M^*_{i,j}}$. It can be seen that K_x is valid

$$\begin{aligned} K_x &= L^{z_x} \cdot \prod_{i \in X} \prod_{j=1,...,n^*} (g^{(a^j/b_i) \cdot r} \cdot \prod_{k=1,...,n^*, k \neq j} (g^{a^{q+1+j-k}/b_i})^{w_k})^{M^*_{i,j}} \\ &= L^{z_x} \cdot \prod_{i \in X} \prod_{j=1,...,n^*} (g^{(a^j/b_i) \cdot r} \cdot \prod_{k=1,...,n^*, k \neq j} (g^{a^{q+1+j-k}/b_i})^{w_k})^{M^*_{i,j}} \cdot \prod_{i \in X} \prod_{j=1,...,n^*} (g^{a^{q+1}/b_i})^{w_j \cdot M^*_{i,j}} \\ &= (g^r \cdot \prod_{i=1,...,n^*} g^{a^{q+1-i} \cdot w_i})^{z_x} \cdot \prod_{i \in X} \prod_{j=1,...,n^*} (g^{(a^j/b_i) \cdot r} \cdot \prod_{k=1,...,n^*} (g^{a^{q+1+j-k}/b_i})^{w_k})^{M^*_{i,j}} \\ &= (g^{z_x} \cdot \prod_{i \in X} g^{a \cdot M^*_{i,1}/b_i + a^2 \cdot M^*_{i,2}/b_i + \cdots + a^{n^*} \cdot M^*_{i,n^*}/b_i})^{(r + w_1 \cdot a^q + \cdots + w_{n^*} \cdot a^{q-n^*+1})} \\ &= \delta_{2,x}^{(r + w_1 \cdot a^q + \cdots + w_{n^*} \cdot a^{q-n^*+1})} = \delta_{2,x}^t = H_3(x)^t, \end{aligned}$$

where X is the set of all i such that $\rho^*(i) = x$. Recall that if $S \not\models (M^*, \rho^*)$, we then have $w \cdot M_i^* = 0$. Thus we have $\prod_{i \in X} \prod_{j=1,...,n^*} (g^{a^{q+1}/b_i})^{w_j \cdot M^*_{i,j}} = g^{a^{q+1} \cdot (\sum_{i \in X} \sum_{j=1,...,n^*} w_j \cdot M^*_{i,j}/b_i)} = g^0 = 1$. Finally, $\mathcal{C}$ adds the tuple (S, sk_S) to SK^{List} and returns sk_S to $\mathcal{A}$.

(b) *Re-encryption key extraction oracle* $\mathcal{O}_{rk}(S, (M', \rho'))$: if $(S,\ (M', \rho'),\ \delta,\ \beta',\ rk_{S \to (M',\rho')},\ *,\ 0,\ 1) \in RK^{List}$, $\mathcal{C}$ returns $rk_{S \to (M',\rho')}$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ works as follows.

- If $S \models (M^*, \rho^*) \wedge (S', sk_{S'}) \in SK^{List}$ (for any $S' \models (M', \rho')$), $\mathcal{C}$ outputs a random bit in $\{0, 1\}$ and aborts the simulation (due to the restrictions defined in Definition 4).
- If $S \models (M^*, \rho^*) \wedge (S', sk_{S'}) \notin SK^{List}$ (for any $S' \models (M', \rho')$), $\mathcal{C}$ checks whether $(S,\ (M', \rho'),\ \delta,\ \beta',\ rk_{S \to (M',\rho')},\ 1,\ 1,\ 0) \in RK^{List}$. If yes, $\mathcal{C}$ returns $rk_{S \to (M',\rho')}$ to $\mathcal{A}$ and resets $tag_2 = 0, tag_3 = 1$. Otherwise, $\mathcal{C}$ first chooses $\theta, \sigma \in_R \mathbb{Z}_p^*$, $\beta', \delta \in_R \{0, 1\}^k$, $\bar{K} \in_R \mathbb{G}$. $\mathcal{C}$ then sets $rk_1 = \bar{K} \cdot g_1^{\theta}$, $rk_2 = g^{\theta}$, $rk_3 = g^{\sigma}$, $R_x = \delta_{2,x}^{\sigma}$, and constructs rk_4 as in the real scheme, where $\delta_{2,x}$ is the output of issuing x to H_3, $x \in S$. Finally, $\mathcal{C}$ returns $rk_{S \to (M',\rho')} = (S, rk_1, rk_2, rk_3, rk_4, R_x)$ to $\mathcal{A}$, and adds $(S,\ (M', \rho'),\ \delta,\ \beta',\ rk_{S \to (M',\rho')},\ 1,\ 0,\ 1)$ to RK^{List}.

[14] Such a vector w must exist by the convention of an LSSS. Please refer to the discussion in [26].

- Otherwise, if $(S,\ (M',\rho'),\ \delta,\ \beta',\ rk_{S\to(M',\rho')},\ 0,\ 1,\ 0) \in RK^{List}$, $\mathcal{C}$ returns $rk_{S\to(M',\rho')}$ to $\mathcal{A}$, and resets $tag_2 = 0, tag_3 = 1$. Otherwise, $\mathcal{C}$ first constructs the private key sk_S for the attribute set S as step (a). $\mathcal{C}$ further generates $rk_{S\to(M',\rho')}$ as in the real scheme, returns the re-encryption key to $\mathcal{A}$ and adds $(S,\ (M',\rho'),\ \delta,\ \beta',\ rk_{S\to(M',\rho')},\ 0,\ 0,\ 1)$ to RK^{List}.

(c) *Re-encryption oracle* $\mathcal{O}_{re}(S,(M',\rho'),C_{(M,\rho)})$: $\mathcal{C}$ verifies whether Eq. (1) holds. If not (i.e. indicating either the ciphertext $C_{(M,\rho)}$ is invalid or $S \nvDash (M,\rho)$), $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ proceeds.

- If $S \models (M^*,\rho^*) \wedge (S',sk_{S'}) \in SK^{List}$ (for any $S' \models (M',\rho')$) does not hold,
  - i. If $S \models (M^*,\rho^*) \wedge (S',sk_{S'}) \notin SK^{List}$, $\mathcal{C}$ first constructs the re-encryption key as the second case of step (b), further re-encrypts $C_{(M,\rho)}$ to $\mathcal{A}$, and finally adds $(S,\ (M',\rho'),\ \delta,\ \beta',\ rk_{S\to(M',\rho')},\ 1,\ 1,\ 0)$, $(S,(M',\rho'),C^R_{(M',\rho')},0,1,0)$ to RK^{List}, RE^{List}, respectively.
  - ii. Otherwise, $\mathcal{C}$ first constructs the re-encryption key as the third case of step (b), further re-encrypts $C_{(M,\rho)}$ to $\mathcal{A}$, and finally adds $(S,\ (M',\rho'),\ \delta,\ \beta',\ rk_{S\to(M',\rho')},\ 0,\ 1,\ 0)$, $(S,\ (M',\rho'),\ C^R_{(M',\rho')},\ 1,\ 0,\ 0)$ to RK^{List}, RE^{List}, respectively.
- Otherwise, $\mathcal{C}$ checks whether $(m,\beta,s) \in H_1^{List}$ such that $A_3 = g_1^s$. If no such tuple exists, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ checks whether $(S,\ (M',\rho'),\ \delta,\ \beta',\ \perp,\ \perp,\ 1,\ \perp) \in RK^{List}$, where $S \models (M^*,\rho^*)$. If no, $\mathcal{C}$ chooses $\beta', \delta \in_R \{0,1\}^k$, generates $rk_4 = C_{(M',\rho')}$ (to hide δ and β') as in the real scheme, and constructs $A_4 = (e(g^a,g^{a^q}) \cdot e(g,g^{\alpha'}))^{s\cdot\xi_2}$, where $\xi_2 = H_5(\delta)$. Finally, $\mathcal{C}$ returns $C^R_{(M',\rho')} = (S,\ (M,\rho),\ A_1,\ A_3,\ (B_1,C_1),\ ...,\ (B_l,C_l),\ D,\ A_4,\ rk_4)$ to $\mathcal{A}$, and adds $(S,\ (M',\rho'),\ \delta,\ \beta',\ \perp,\ \perp,\ 1,\ \perp)$, $(S,\ (M',\rho'),\ C^R_{(M',\rho')},\ 0,\ 0,\ 1)$ to RK^{List}, RE^{List}, respectively.

(d) *Original ciphertext decryption oracle* $\mathcal{O}_{d2}(S,C_{(M,\rho)})$: $\mathcal{C}$ verifies whether Eq. (1) holds. If not (i.e. indicating either the ciphertext is invalid or $S \nvDash (M,\rho)$), $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ proceeds.

- If $(S,sk_S) \in SK^{List}$ (for any $S \models (M,\rho)$), $\mathcal{C}$ recovers m as in the real scheme using sk_S.
- Otherwise, $\mathcal{C}$ checks whether $(m,\beta,s) \in H_1^{List}$ and $(R,\delta_1) \in H_2^{List}$ such that $A_3 = g_1^s$, $A_1 = (m||\beta) \oplus \delta_1$ and $R = e(g,g)^{\alpha\cdot s}$. $\mathcal{C}$ outputs $\perp$ if no such tuples exist, and outputs m otherwise.

(e) *Re-encrypted ciphertext decryption oracle* $\mathcal{O}_{d1}(S',C^R_{(M',\rho')})$: $\mathcal{C}$ first checks whether there are tuples (δ,β',s') and (m,β,s) in H_1^{List} such that $A_2' = g^{s'}$ and $A_3 = g_1^s$. If not, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ verifies whether Eq. (2) holds. If not (i.e. indicating either rk_4 is invalid or $S' \nvDash (M',\rho')$), $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ proceeds.

- If $(S,\ (M',\rho'),\ \delta,\ \beta',\ rk_{S\to(M',\rho')},\ 1,\ 0,\ 1) \in RK^{List} \vee (S,\ (M',\rho'),\ C^R_{(M',\rho')},\ 0,\ 1,\ 0) \in RE^{List}$, $\mathcal{C}$ checks

$$e(\prod_{i\in I'} B_i'^{w_i'}, g) \stackrel{?}{=} e(A_2', g^a) \cdot \prod_{i\in I'} (e(C_i'^{-1}, H_3(\rho'(i))^{w_i'})), \tag{3}$$

  where with knowledge of M' and I' ($I' \subset \{1,...,l'\}$ and $I' = \{i : \rho'(i) \in S'\}$), $\mathcal{C}$ can find a vector $w' = \{w_i' \in \mathbb{Z}_p^*\}_{i\in I'}$ such that $\sum_{i\in I'} w_i' \cdot \lambda_i' = s'$. If Eq. (3) does not hold, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ reconstructs $A_2 = g^s$ with knowledge of s and then verifies Eq. (1). If the equation does not hold, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ recovers the random re-encryption key $rk_{S\to(M',\rho')} = (S, rk_1, rk_2, rk_3, rk_4, R_x)$ from RK^{List}, and checks the validity of A_4 as $A_4 \stackrel{?}{=} \frac{e(A_2,rk_1)/e(A_3,rk_2)}{(\prod_{i\in I}(e(B_i,rk_3)\cdot e(C_i,R_{\rho(i)}))^{w_i})}$, where I and w_i are defined in *ReEnc* on page 5. If the above equation does not hold, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ checks whether $(R,\delta_1) \in H_2^{List}$ such that $A_1 = (m||\beta) \oplus \delta_1$ and $R = e(g,g)^{\alpha\cdot s}$. If no such tuple exists, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ returns m to $\mathcal{A}$. Note that $\mathcal{C}$ can tell the derivatives of the challenge ciphertext via the above manner.
- Otherwise,

i. If $(S', sk_{S'}) \in SK^{List}$, $\mathcal{C}$ recovers m as in the real scheme using $sk_{S'}$.
ii. Otherwise, $\mathcal{C}$ checks whether Eq. (2) and Eq. (3) hold. If not, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ checks whether $(R, \delta_1) \in H_2^{List}$ such that $A_1 = (m||\beta) \oplus \delta_1$, $R = e(g,g)^{\alpha \cdot s}$, and verifies whether $A_4 = e(g,g)^{\alpha \cdot s \cdot \xi_2}$ and $D = H_4(A_1, A_3, (B_1, C_1), ..., (B_l, C_l), (M, \rho))^s$ hold, where $\xi_2 = H_5(\delta)$. If no such tuple exists and the equations do not hold, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ returns m to $\mathcal{A}$.

4. **Challenge.** $\mathcal{A}$ outputs m_0, m_1 to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in_R \{0,1\}$ and responds as follows.
   (a) For each row i of M^*, set $x^* = \rho^*(i)$, and issue an H_3 query on x^* to obtain the tuple (x^*, z_{x^*}, δ_{2,x^*}). Like [26] (the challenger is able to choose the secret splitting), choose y'_2, ..., y'_{n^*} and share the seccret using the vector $v = (s, s \cdot a + y'_2, s \cdot a^2 + y'_3, ..., s \cdot a^{n-1} + y'_{n^*}) \in \mathbb{Z}_p^{n^*}$. Choose r'_1, ..., $r'_{l^*} \in_R \mathbb{Z}_p^*$, for all $i \in \{1, ..., l^*\}$, denote R_i as the set of all $i \neq k$ such that $\rho^*(i) = \rho^*(k)$. Set

$$B_i^* = \delta_{2,x^*}^{-r'_i} \cdot (\prod_{j=2,...,n^*} g^{a \cdot M_{i,j}^* \cdot y'_j}) \cdot g^{b_i \cdot s \cdot (-z_{x^*})} \cdot (\prod_{k \in R_i} \prod_{j=1,...,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M_{k,j}^*})^{-1}, C_i^* = g^{r'_i + s \cdot b_i}.$$

   (b) Choose $\beta^* \in_R \{0,1\}^k$, $A_1^* \in_R \{0,1\}^{2k}$, implicitly define $H_2(T \cdot e(g^s, g^{\alpha'})) = A_1^* \oplus (m_b||\beta^*)$, and set $A_2^* = g^s$, $A_3^* = (g^s)^\gamma$.
   (c) Issue an H_4 query on (A_1^*, A_3^*, (B_1^*, C_1^*), ..., $(B_{l^*}^*, C_{l^*}^*)$, (M^*, ρ^*)) to obtain the tuple (A_1^*, A_3^*, (B_1^*, C_1^*), ..., $(B_{l^*}^*, C_{l^*}^*)$, (M^*, ρ^*), ξ_1^*, δ_3^*), and define $D^* = (g^s)^{\xi_1^*}$.
   (d) Output the challenge ciphertext $C_{(M^*,\rho^*)}^* = ((M^*, \rho^*), A_1^*, A_2^*, A_3^*, (B_1^*, C_1^*), ..., (B_{l^*}^*, C_{l^*}^*), D^*)$ to $\mathcal{A}$.

   If $T = e(g,g)^{a^{q+1} \cdot s}$, $C_{(M^*,\rho^*)}^*$ is a valid ciphertext. Implicitly letting $H_1(m_b, \beta^*) = s$ and $r_i = r'_i + s \cdot b_i$, one can verify that

$$A_1^* = A_1^* \oplus (m_b||\beta^*) \oplus (m_b||\beta^*) = H_2(T \cdot e(g^s, g^{\alpha'})) \oplus (m_b||\beta^*) = H_2(e(g,g)^{\alpha \cdot s}) \oplus (m_b||\beta^*),$$
$$A_2^* = g^s, A_3^* = (g^s)^\gamma = (g^\gamma)^s = g_1^s, D^* = (g^s)^{\xi_1^*} = (g^{\xi_1^*})^s = H_4(A_1^*, A_3^*, (B_1^*, C_1^*), ..., (B_{l^*}^*, C_{l^*}^*), (M^*, \rho^*))^s,$$
$$B_i^* = \delta_{2,x^*}^{-r'_i}(\prod_{j=2,...,n^*} (g^a)^{M_{i,j}^* \cdot y'_j})(g^{b_i \cdot s})^{-z_{x^*}}(\prod_{k \in R_i} \prod_{j=1,...,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M_{k,j}^*})^{-1}$$
$$= \delta_{2,x^*}^{-r'_i}(\prod_{j=2,...,n^*} (g^a)^{M_{i,j}^* \cdot y'_j})(\prod_{j=1,...,n^*} g^{a^j \cdot s \cdot M_{i,j}^*})(\prod_{j=1,...,n^*} g^{a^j \cdot s \cdot M_{i,j}^*})^{-1}(g^{b_i \cdot s})^{-z_{x^*}}(\prod_{k \in R_i} \prod_{j=1,...,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M_{k,j}^*})^{-1}$$
$$= \delta_{2,x^*}^{-r'_i} g^{a\lambda_i}(\prod_{j=1,...,n^*} g^{a^j \cdot s \cdot M_{i,j}^*})^{-1}(g^{b_i \cdot s})^{-z_{x^*}}(\prod_{k \in R_i} \prod_{j=1,...,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M_{k,j}^*})^{-1}$$
$$= g^{a\lambda_i} g^{z_{x^*} \cdot (-r'_i)} g^{b_i \cdot s \cdot (-z_{x^*})} (\prod_{i \in X} g^{a \cdot M_{i,1}^*/b_i + a^2 \cdot M_{i,2}^*/b_i + \cdots + a^{n^*} \cdot M_{i,n^*}^*/b_i})^{-r'_i} (\prod_{j=1,...,n^*} g^{a^j \cdot s \cdot M_{i,j}^*})^{-1}$$
$$\cdot (\prod_{k \in R_i} \prod_{j=1,...,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M_{k,j}^*})^{-1}$$
$$= g^{a\lambda_i} \delta_{2,x^*}^{-r'_i - s \cdot b_i} = g^{a\lambda_i} \delta_{2,x^*}^{-r_i} = g^{a\lambda_i} H_3(x^*)^{-r_i} = g^{a\lambda_i} H_3(\rho^*(i))^{-r_i}, C_i^* = g^{r'_i + s \cdot b_i} = g^{r_i}.$$

   However, if $T \in_R \mathbb{G}_T$, the challenge ciphertext is independent of the bit b in the view of $\mathcal{A}$.
5. **Query Phase II.** Same as Query Phase I but with the constraints defined in Definition 4.
6. **Guess.** $\mathcal{A}$ outputs a guess bit $b' \in \{0,1\}$. If $b' = b$, $\mathcal{C}$ outputs 1 (i.e. deciding $T = e(g,g)^{a^{q+1} \cdot s}$); otherwise, $\mathcal{C}$ outputs 0 (i.e. deciding $T \in_R \mathbb{G}_T$).

We first analyze the simulations of the random oracles. Clearly, the simulations of the oracles are perfect except H_1 and H_2. Let H_1^* and H_2^* be the events that $\mathcal{A}$ has queried (m_b, β^*) to H_1 and $R^* = e(g,g)^{\alpha \cdot s}$ to H_2 before the challenge phase, respectively, where $\beta^*, b \in \{0,1\}$ are chosen by $\mathcal{C}$ in the challenge phase. Except for the cases above, the simulations of H_1 and H_2 are perfect. We denote

by $Adv^{TCR}_{H_1^*,\mathcal{A}}$ the probability of $\mathcal{A}$ in successfully querying (m_b, β^*) from H_1 before the challenge phase. Similarly, we have $Adv^{TCR}_{H_2^*,\mathcal{A}}$.

In the simulation of the private key generation, the responses to $\mathcal{A}$ are perfect. As to the simulation of the re-encryption key queries, the responses to $\mathcal{A}$ are also perfect except for the case where the re-encryption key is randomly generated. It can be seen that rk_1, rk_2, rk_3 and R_x (which are generated by $\mathcal{C}$) can take the form of the corresponding components of the valid re-encryption key, respectively. Hence, the indistinguishability between the random re-encryption key and the valid one depends on the indistinguishability between the encryption generated by $\mathcal{C}$ and the one constructed in the real scheme. If $\mathcal{A}$ can distinguish the encryptions above, then $\mathcal{C}$ can break the decisional q-parallel BDHE problem using $\mathcal{A}$. As for the simulation given in the challenge phase, it is perfect as well.

In the simulation of the re-encryption queries, the responses to $\mathcal{A}$ are perfect with the exception that $\mathcal{A}$ submits a valid original ciphertext which is generated without issuing the query to H_1. We denote by $Pr[ReEncErr]$ the probability of the above exception. Then, we have $Pr[ReEncErr] \leq \frac{q_{re}}{p}$, where q_{re} is the total number of re-encryption queries.

In the simulation of decryption queries, it might be possible that $\mathcal{C}$ cannot provide a decryption for a valid ciphertext. Suppose $\mathcal{A}$ can generate a valid ciphertext without querying $e(g,g)^{\alpha \cdot s}$ to H_2, where $s = H_1(m, \beta)$. Let $valid$ be the event that the original ciphertext or the re-encrypted ciphertext is valid, $QueryH_1$ be the event that $\mathcal{A}$ has queried (m, β) to H_1 and $QueryH_2$ be the event that $\mathcal{A}$ has queried $e(g,g)^{\alpha \cdot s}$ to H_2. From the simulation, we have

$$Pr[valid|\neg QueryH_2] \leq Pr[QueryH_1|\neg QueryH_2] + Pr[valid|\neg QueryH_1 \wedge \neg QueryH_2] \leq \frac{q_{H_1}}{2^{2k}} + \frac{1}{p}$$

and $Pr[valid|\neg QueryH_1] \leq \frac{q_{H_2}}{2^{2k}} + \frac{1}{p}$, where q_{H_1} and q_{H_2} are the maximum number of random oracle queries to H_1 and H_2. Let $Pr[DecErr]$ be the probability that the event $valid|(\neg QueryH_1 \vee \neg QueryH_2)$ occurs, then we have $Pr[DecErr] \leq (\frac{q_{H_1}+q_{H_2}}{2^{2k}} + \frac{2}{p}) \cdot (q_{d1} + q_{d2})$, where q_{d2} and q_{d1} denote the total numbers of original ciphertext decryption queries and re-encrypted ciphertexts decryption queries.

Let Bad denote the event that $(H_1^*|\neg H_2^*) \vee H_2^* \vee ReEncErr \vee DecErr$. Then we have

$$\begin{aligned}\epsilon_1 &= |Pr[b' = b] - \frac{1}{2}| \leq \frac{1}{2} Pr[Bad] = \frac{1}{2} Pr[(H_1^*|\neg H_2^*) \vee H_2^* \vee ReEncErr \vee DecErr] \\ &\leq \frac{1}{2}(Adv^{TCR}_{H_2^*,A} + \frac{q_{H_1} + (q_{H_1} + q_{H_2}) \cdot (q_{d_1} + q_{d_2})}{2^{2k}} + \frac{2(q_{d_1} + q_{d_2}) + q_{re}}{p}).\end{aligned}$$

Therefore, $Adv^{D-q-parallelBDHE}_{\mathcal{A}} \geq \frac{1}{q_{H_2}}(Adv^{TCR}_{H_2^*,A}) \geq \frac{1}{q_{H_2}}(2\epsilon_1 - \frac{q_{H_1}+(q_{H_1}+q_{H_2})\cdot(q_{d_1}+q_{d_2})}{2^{2k}} - \frac{2(q_{d_1}+q_{d_2})+q_{re}}{p})$.

From the simulation, the running time of $\mathcal{C}$ is bound by

$$\begin{aligned}t' \leq\ & t + O(1)(q_{H_1} + q_{H_2} + q_{H_3} + q_{H_4} + q_{H_5} + q_{H_6} + q_{sk} + q_{rk} + q_{re} + q_{d2} + q_{d1}) \\ & + t_e(q_{sk}O(n^{*2}) + (q_{rk} + q_{re})O(f) + (q_{d2} + q_{d1})O(l) + q_{H_1}(q_{re} + q_{d2} + q_{d1})O(1)) \\ & + t_p((q_{re} + q_{d2} + q_{d1})O(l)),\end{aligned}$$

where q_{H_i} denotes the total number of random oracle queries to H_i ($i \in \{1,2,3,4,5,6\}$), q_{sk} and q_{rk} denote the total numbers of private key extraction queries and re-encryption key extraction queries, t_e denotes the running time of an exponentiation in group $\mathbb{G}$, t_p denotes the running time of a pairing in group $\mathbb{G}_T$, t is the running time of $\mathcal{A}$, l is the number of rows of matrix.

This completes the proof of Theorem 2. □

B.2 Proof of Theorem 3

Proof. Suppose there exists an adversary $\mathcal{A}$ who can break the IND-sAS-CCA-Re security of our scheme. We then construct a reduction algorithm $\mathcal{C}$ to plays the decisional q-parallel BDHE problem.

$\mathcal{C}$ takes in $(p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow BSetup(1^k)$ and a q-parallel BDHE instance y and $T \in \mathbb{G}_T$, where T is either equal to $e(g,g)^{a^{q+1} \cdot s}$ or to $T' \in_R \mathbb{G}_T$.

1. **Initialization.** Same as the proof of Theorem 2.
2. **Setup.** Same as the proof of Theorem 2.
3. **Query Phase I.** Same as the proof of Theorem 2 but with constraints defined in Definition 5.
4. **Challenge.** $\mathcal{A}$ outputs (M, ρ), m_0 and m_1 to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in_R \{0,1\}$ and responds as follows.
   (a) Run $C_{(M,\rho)} \leftarrow Enc((M,\rho), m_b)$ as in the real scheme, and output $((M,\rho), A_1, A_3, (B_1, C_1), ..., (B_l, C_l), D)$. Here the component A_2 is unnecessarily output. Note that M is an $l \times n$ matrix.
   (b) Find an attribute set S such that $S \models (M,\rho)$[15], and choose $\beta'^*, \delta^* \in_R \{0,1\}^k$. Issue an H_5 query on δ^* to obtain ξ_2^*. Note that in step (a) the query (m_b, β) must be issued to H_1 such that the tuple (m_b, β, s') is already stored in H_1^{List}, where $\beta \in \{0,1\}^k, s' \in \mathbb{Z}_p^*$. Then recover (m_b, β, s') from H_1^{List}, and set $A_4^* = (e(g^a, g^{a^q}) \cdot e(g, g^{\alpha'}))^{s' \cdot \xi_2^*}$.
   (c) For each row i of M^* (an $l^* \times n^*$ matrix), set $x^* = \rho^*(i)$, issue an H_3 query on x^* to obtain the tuple $(x^*, z_{x^*}, \delta_{2,x^*})$. Choose $y'_2, ..., y'_{n^*}, r'_1, ..., r'_{l^*} \in_R \mathbb{Z}_p^*$, for all $i \in \{1, ..., l^*\}$, denote R_i as the set of all $i \neq k$ such that $\rho^*(i) = \rho^*(k)$. Set

$$B_i'^* = \delta_{2,x^*}^{-r'_i} \cdot \Big(\prod_{j=2,...,n^*} g^{a \cdot M^*_{i,j} \cdot y'_j}\Big) \cdot g^{b_i \cdot s \cdot (-z_{x^*})} \cdot \Big(\prod_{k \in R_i} \prod_{j=1,...,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M^*_{k,j}}\Big)^{-1}, C_i'^* = g^{r'_i + s \cdot b_i}.$$

   (d) Choose $A_1'^* \in_R \{0,1\}^{2k}$, implicitly define $H_2(T \cdot e(g^s, g^{\alpha'})) = A_1'^* \oplus (\delta^* || \beta'^*)$, and set $A_2'^* = g^s$.
   (e) Issue an H_6 query on $(A_1'^*, A_2'^*, (B_1'^*, C_1'^*), ..., (B_{l^*}'^*, C_{l^*}'^*), S, (M^*, \rho^*))$ to obtain the tuple $(A_1'^*, A_2'^*, (B_1'^*, C_1'^*), ..., (B_{l^*}'^*, C_{l^*}'^*), S, (M^*, \rho^*), \xi_3^*, \delta_4^*)$, and define $D'^* = (g^s)^{\xi_3^*}$.
   (f) Output the challenge ciphertext $C^{R*}_{(M^*,\rho^*)} = (S, (M,\rho), (M^*,\rho^*), A_1, A_3, (B_1, C_1), ..., (B_l, C_l), D, A_4^*, A_1'^*, A_2'^*, (B_1'^*, C_1'^*), ..., (B_{l^*}'^*, C_{l^*}'^*), D'^*)$ to $\mathcal{A}$.

   If $T = e(g,g)^{a^{q+1} \cdot s}$, $C^{R*}_{(M^*,\rho^*)}$ is a valid ciphertext. Clearly, the components corresponding to $C_{(M,\rho)}$ are valid. Since $C_{(M,\rho)}$ is re-encrypted to $C^{R*}_{(M^*,\rho^*)}$ under a valid re-encryption key $rk_{S \to (M^*,\rho^*)}$ $(S \models (M,\rho))$, the re-encryption must be valid, i.e. the construction of A_4^* is valid. With the same analysis technique given in the proof of Theorem 2, it is not difficult to see that the rest of components are valid as well. If $T \in_R \mathbb{G}_T$, the challenge ciphertext is independent of the bit b in the view of $\mathcal{A}$.
5. **Query Phase II.** Same as Query Phase I but with the constraints defined in Definition 5.
6. **Guess.** $\mathcal{A}$ outputs a guess bit $b' \in \{0,1\}$. If $b' = b$, $\mathcal{C}$ outputs 1 (i.e. deciding $T = e(g,g)^{a^{q+1} \cdot s}$); otherwise, $\mathcal{C}$ outputs 0 (i.e. deciding $T \in_R \mathbb{G}_T$).

The probability analysis is almost the same as the one given in Appendix B.1 except that we should take the even H_5^* into account when analyzing the event Bad, where H_5^* denotes the event that $\mathcal{A}$ has queried δ^* to H_5 before the challenge phase. From the simulation we have $Adv_{\mathcal{A}}^{D-q-parallelBDHE} \geq \frac{1}{q_{H_2}}(Adv^{TCR}_{H_2^*,A}) \geq \frac{1}{q_{H_2}}(2\epsilon_2 - \frac{q_{H_1} + (q_{H_1} + q_{H_2}) \cdot (q_{d_1} + q_{d_2})}{2^{2k}} - \frac{q_{H_5}}{2^k} - \frac{2(q_{d_1} + q_{d_2}) + q_{re}}{p})$. The running time of $\mathcal{C}$ is identical to that given in Appendix B.1. We hence omit the details.

This completes the proof of Theorem 3. □

[15] Note that it is possible that S can be found in SK^{List}.