1-1-2014

Empowering personal health records with cloud computing: how to encrypt with forthcoming fine-grained policies efficiently

Clementine Gritti
University of Wollongong, cjpg967@uowmail.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Thomas Plantard
University of Wollongong, thomaspl@uow.edu.au

Kaitai Liang
City University of Hong Kong

Duncan S. Wong
City University of Hong Kong, dwong@uow.edu.au

Empowering personal health records with cloud computing: how to encrypt with forthcoming fine-grained policies efficiently

Abstract

The issue of empowering patients to be well informed with regards to their health records has been well accepted in the community, which is known as the Personal Health Record (PHR). PHR has been believed as the solution for better management of an individual's health, and as the tool that will empower the patient in correlation with healthcare providers through the ability to provide his/her own medical history. In this work, we aim to take one step further by equipping patients with the ability to "control" the access to their PHR efficiently and easily, by incorporating the emerging cloud technology. Specifically, we aim to provide the patients with the luxury of using the power of the cloud to conduct the outsourced work efficiently. To realize this, we present the notion of online/offline ciphertext-policy attribute-based proxy re-encryption scheme, which is very useful primitive in empowering personal health records in cloud computing. We present such a notion as well as a set of security requirements. More specifically, we define two security models covering both outsider and insider attacks. Furthermore, we present a concrete construction of such a scheme, and prove that it is secure under the well known complexity assumptions and following our security models.

Keywords

Ciphertext-Policy Attribute-Based Encryption, Proxy Re-Encryption, Online/Offline encryption, CCA security

Disciplines

Engineering | Science and Technology Studies

Publication Details

Gritti, C., Susilo, W., Plantard, T., Liang, K. and Wong, D. (2014). Empowering personal health records with cloud computing: how to encrypt with forthcoming fine-grained policies efficiently. Journal of Wireless Mobile Networks, Ubiquitous Computing and Dependable Applications, 5 (4), 3-28.

Empowering Personal Health Records with Cloud Computing

How to encrypt with forthcoming fine-grained policies efficiently

Clémentine Gritti
Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
cjpg967@uowmail.edu.au

Willy Susilo
Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
wsusilo@edu.au

Thomas Plantard
Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
thomaspl.edu.au

Kaitai Liang
Department of Information and Computer Science
Aalto University, Finland
kaitai.liang@aalto.fi

Duncan S. Wong
Department of Computer Science
City University of Hong Kong, Hong Kong
duncan@cityu.edu.hk

Abstract

The issue of empowering patients to be well informed with regards to their health records has been well accepted in the community, which is known as the Personal Health Record (PHR). PHR has been believed as the solution for better management of an individual's health, and as the tool that will empower the patient in correlation with healthcare providers through the ability to provide his/her own medical history. In this work, we aim to take one step further by equipping patients with the ability to "control" the access to their PHR efficiently and easily, by incorporating the emerging cloud technology. Specifically, we aim to provide the patients with the luxury of using the power of the cloud to conduct the outsourced work efficiently. To realize this, we present the notion of online/offline ciphertext-policy attribute-based proxy re-encryption scheme, which is very useful primitive in empowering personal health records in cloud computing. We present such a notion as well as a set of security requirements. More specifically, we define two security models covering both outsider and insider attacks. Furthermore, we present a concrete construction of such a scheme, and prove that it is secure under the well known complexity assumptions and following our security models.

Keywords Ciphertext-Policy Attribute-Based Encryption, Proxy Re-Encryption, Online/Offline encryption, CCA security.

1 Introduction

The focus of healthcare has been shifted from healthcare providers' paternalistic approach to the consumer oriented approach. It is noted that consumers that are well informed about their illnesses tend to understand and to follow instructions and to ask more insightful questions. The PHR (Personal Health Record) could be seen as the solution for better management of an individual's health, and as the tool that will empower the patient in correlation with healthcare providers through the ability to provide his/her

own medical history. Patient access to their own record is important as respecting clinicians should respect patient's autonomy and disclose to their patients relevant information regarding their diagnosis, prognosis or the implications of diagnostic tests. In this work, we are motivated to empower PHR with the emerging cloud computing solution. Especially, we would like to equip patients with the ability to "control" the access to their PHR efficiently and easily, while the patients are away from their desktop computing solution.

To motivate this work, let us consider the following scenario. The blood test at the medical institute has just collected a patient's, Alice's, blood test sample at the medical institute. As the result of the blood test is now available, the medical institute will then encrypt it and send it to their hospital's private cloud. We note that the cloud will not be able to decrypt this ciphertext without Alice's private key, which is inaccessible by the cloud. Technically, we can view this ciphertext as an *intermediate ciphertext*. Alice is aware that her blood result test is available in the cloud. Now, Alice can decide which physician at the hospital that she would like to discuss this result with, and therefore, she will need to enable the access control to that particular physician. As an example, she can further restrict the time of the access as well, i.e. during the appointment time. What Alice needs to do is merely to create a re-encryption key using her own private key, and provides it to the cloud. Note that this re-encryption key will not enable the cloud to decrypt the message, but rather it will just enable the cloud to "re-encrypt" the ciphertext and it will be decryptable by the appointed physician. As an example, the policy that Alice constructs can be something like $\{$Dr. Brown $\wedge$ Monday 15/09/2014 at 1:00pm to 2:00pm$\}$, where 1:00pm to 2:00pm indicates the time of her appointment with Dr. Brown. Subsequently, the cloud can re-encrypt the intermediate ciphertext under Alice's policy and keeps it securely in the cloud. Only Dr. Brown will be able to decrypt this result in the appointed time range. This scenario is depicted in Fig. 1.

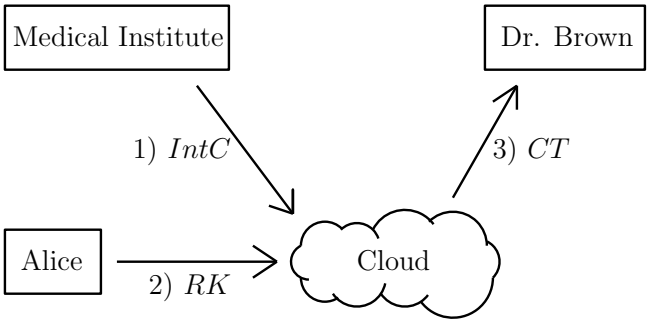

Figure 1: The medical institute generates the intermediate cipertext *IntC* and sends it to the cloud. The patient Alice generates the re-encryption key *RK* based on her policy and sends it to the cloud. The cloud generates the final ciphertext *CT* using *IntC* and *RK* and sends it to Dr. Brown.

In order to realize the solution for this problem, we propose the notion of online/offline ciphertext attribute-based proxy encryption (OO-CP-AB-PRE). This is a new cryptographic notion, which may be also of interest for other applications.

One may notice that the recent notion of Online/Offline Attribute-based Encryption (OO-ABE) [15] is closely related to this scenario, and it might be a candidate as a solution to this problem. Unfortunately, the notion of OO-ABE is insufficient to solve the above scenario. If we employ an OO-ABE in the above scenario, then the notion of cloud is redundant, as Alice would need to re-encrypt the ciphertext (i.e. the intermediate ciphertext in the above context) to provide the access to Dr. Brown. We argue that this is a cumbersome process as this process may not be feasible to be conducted by Alice, who may just be equipped with a smartphone to do so. Re-encrypting the whole intermediate ciphertext might be too costly to be done in a mobile device, and therefore we need to "outsource" this activity to the cloud.

Furthermore, in practice, if OO-ABE is used in this scenario, we will still require the involvement of the cloud as the private cloud of the hospital that will store all the results of the blood tests (and any other medical tests conducted at the medical institute). Hence, the cloud will only be used as a storage and therefore, after the medical institute provides the result of the blood test, it will then need to store the encrypted result in the cloud. When Alice requires to re-encrypt this ciphertext to enable Dr. Brown to read it, then she will need to download the whole ciphertext entirely prior to conducting the required re-encryption. Due to these reasons, OO-ABE is insufficient to solve the above problem.

1.1 Our Contributions

We intend to propose the notion of Online/Offline Ciphertext Attribute-based Proxy Re-Encryption (OO-CP-AB-PRE) to directly solve the above scenario. We offer the definitions and security models for such a cryptographic primitive. In particular, we offer a selective access structure and chosen ciphertext security game for the OO-CP-AB-PRE system that covers the outsider attacks. We also show that this system is selectively collusion resistant that covers the insider attacks. Based on the new techniques for Online/Offline Attribute-Based Encryption (ABE) systems from [15] and the methods for Ciphertext-Policy Attribute-Based Proxy Re-Encryption (CP-AB-PRE) systems used in [20], we provide the first contruction of CCA secure single-hop unidirectional CP-AB-PRE supporting any monotonic access policy and dealing with Online/Offline encryption. In this setting, the ciphertext is associated with an access policy represented as a Linear Secret Sharing Scheme (LSSS) matrix and the private key is attached to a set of attributes. Our scheme is proven secure in the random oracle model under the "$q-1$" assumption [28].

1.2 Related Work

Attribute-Based Encryption (ABE). Sahai and Waters [29] introduced the notion of Attribute-Based Encryption (ABE). Goyal et al. [13] implemented the first Key-Policy ABE (KP-ABE) such that the ciphertexts are associated to an attribute set and each of the private keys is related to an access policy over the attributes. The concept of an ABE setting with multiple central authorities was addressed in [9, 10]. Constructions that do not consider the issue of collusion resistance can be found in [31, 24, 7]. Lewko and Waters [19] gave the first large universe KP-ABE scheme in composite order groups and in the standard model. Based on the techniques initiated by Okamoto and Takashima [26], Lewko [17] suggested the first large universe KP-ABE scheme in prime order groups. Recently, Rouselakis and Waters [28] proposed two large universe ABE constructions proven selectively secure in the standard model under two "q-type" assumptions.

Ciphertext-Policy ABE. Bethencourt et al. [4] proposed the first Ciphertext-Policy ABE (CP-ABE) such that the ciphertexts are related with an access policy and each of the private keys corresponds to a set of attributes. Afterwards, Cheung and Newport [11] suggested a CP-ABE scheme proven secure and supporting only AND gates over attributes. At the same time, Ostrovsky et al. [27] proposed an ABE system with non-montonic access structures: they achieved to relaize negative attributes. Waters [32] proposed the first fully secure CP-ABE construction. Lewko et al. [18] used dual system encryption techniques to obtain a fully secure CP-ABE system but less efficient than the one from [32]. More recently, Attrapadung et al. [2] offered a CP-ABE scheme with constant-size ciphertexts.

Online/Offline CP-ABE. Even et al. [12] introduced the notion of online/offline cryptography applied for signatures. Later, Shamir and Tauman [30] generalized the method based on chameleon hash functions. Guo et al. [14] applied the concept for Identity-Based Encryption (IBE) and obtained an offline encryption system for IBE. Recently, Hohenberger and Waters [15] developped new techniques for On-

line/Offline ABE encryption and key generation and gave two concret schemes, one for key policy and the other for ciphertext policy.

Proxy Re-Encryption. Mambo and Okamoto [23] introduced the concept of decryption rights delegation. Following the idea, Blaze and al. Blaze et al. [5] gave the formalization of Proxy Re-Encryption (PRE) and gave a seminal bidirection PRE scheme. Later, Ivan and Dodis [16] suggested concrete definitions of bidirectional and unidirectional proxy functions. Ateniese et al. [1] constructed three unidirectional CPA-secure PRE schemes. More recently, Canetti and Hohenberger [8] achieved CCA security for PRE systems.

CP-ABE-PRE. Liang et al. [21] first presented a CP-AB-PRE setting and gave a scheme based on the CP-ABE construction proposed in [11] such that AND gates on positive and negative attributes are supported. Mizuno and Doi [25] offered a hybrid PRE construction such that it can bridge ABE and IBE: ciphertexts generated in the ABE setting can be converted to ciphertexts which can be decrypted in the context of IBE. Later, Luo et al. [22] gave a CP-AB-PRE scheme such that AND gates on multivalued and negative attributes are supported. All the aforementionned papers proposed CPA secure CP-AB-PRE systems. Liang et al. [20] proposed the first CP-AB-PRE construction CCA secure and supporting any monotonic access policy.

1.3 Paper Organization

In the next section, we give the some definitions of tools employed in cryptography. In Sec. 3, we present the formal definition of our Online/Offline Ciphertext-Policy Attribute-Based Proxy Re-Encryption (OO-CP-AB-PRE) system, along with its security models definition. In Sec. 4, we build a concrete OO-CP-AB-PRE scheme using pairings. In Sec. 5, we give the security proofs related to our construction, along with the definitions of the used hardness assumptions. In Sec. 6, we discuss about the computation costs in our scheme and compare them to other schemes. Finally, we conclude our paper in the last section.

2 Preliminaries

2.1 Access Structure [3]

Definition 1. *Let $\{P_1, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{AS} \subseteq 2^{\{P_1,\cdots,P_n\}}$ is monotone if the following implication is satisfied: if $\forall B, C$ such that $B \in \mathbb{AS}$ and $B \subseteq C$, then $C \in \mathbb{AS}$. An (monotone) access structure is a (monotone) collection $\mathbb{AS} \subseteq 2^{\{P_1,\cdots,P_n\}} \setminus \{\emptyset\}$. We define the sets in $\mathbb{AS}$ as the authorized sets (we say also that these sets satisfy $\mathbb{AS}$) and the sets not in $\mathbb{AS}$ as the unauthorized sets (we say also that these sets do not satisfy $\mathbb{AS}$).*

In our context, the role of the parties is taken by the attributes. Thus, the access structure $\mathbb{AS}$ will contain the authorized attribute sets.

2.2 Linear Secret-Sharing Scheme (LSSS) [3]

Definition 2. *Let Π be a secret-sharing scheme over a set of parties P. Π is said linear over $\mathbb{Z}_p$ if:*

1. *The shares for each party form a vector over $\mathbb{Z}_p$.*
2. *There is a $l \times n$ matrix M (the share-generating matrix for Π). For all $i = 1, \cdots, l$, the i-th row of M, written as M_i, is labeled by a party $\rho(i)$, where $\rho : \{1, \cdots, l\} \to P$. Let $\vec{y} = (s, y_2, \cdots, y_n)$ be a*

column vector, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $y_2, \cdots, y_n \in \mathbb{Z}_p$ be randomly chosen. Thus, $M\vec{y}$ is the vector of l shares of the secret s according to Π. The share $(M\vec{y})_i$ belongs to party $\rho(i)$.

A linear secret-sharing scheme satisfies the following *linear reconstruction property*. Let Π be a LSSS for the access structure $\mathbb{AS}$, $S \in \mathbb{AS}$ be any authorized set (S satisfies $\mathbb{AS}$), and $I = \{i : \rho(i) \in S\} \subset \{1, \cdots, l\}$. Therefore, there exist constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ satisfying the following implication: if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} w_i \lambda_i = s$.

Convention

The vector $(1, 0, \cdots, 0)$ is the target vector for any LSSS. Let I be a set of rows of M. If I is an authorized set, then $(1, 0, \cdots, 0)$ is in the span of I. If I is an unanthorized set, then $(1, 0, \cdots, 0)$ is not in the span of I. There is a vector w such that $w \cdot (1, 0, \cdots, 0) = -1$ and $\forall i \in I, w \cdot M_i = 0$.

2.3 Bilinear Map

Definition 3. *Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order $p \in \Theta(2^\lambda)$ (where λ is the security parameter). Let g be a generator of $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map with the following properties:*

1. ***Bilinearity:*** $\forall u, v \in \mathbb{G}, \forall a, b \in \mathbb{Z}_p, e(u^a, v^b) = e(u, v)^{ab}$,
2. ***Non-degeneracy:*** $e(g, g) \neq 1_{\mathbb{G}_T}$.

$\mathbb{G}$ is said to be a bilinear group if the group operation in $\mathbb{G}$ and the bilinear map e are both efficiently computable. We can easily see that e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

3 Definitions

3.1 Online/Offline Ciphertext Attribute-based Proxy Re-Encryption (OO-CP-AB-PRE) Model

The Online/Offline model enables as much precomputation of ciphertext as possible such that the required access policy is unknown.

Definition 4. *Let S be a set of attributes and $\mathbb{AS}$ be an access structure. An Online/Offline Ciphertext-Policy Attribute-Based Proxy Re-Encryption (OO-CP-AB-PRE) scheme for access structure space $\mathscr{AS}$, consists of the following six algorithms:*

1. ***Setup***$(\lambda, \mathscr{U}) \to (PK, MSK)$*: on input a security parameter $\lambda \in \mathbb{N}$ and an attribute universe $\mathscr{U}$, output the public parameters PK and a master secret key MSK.*
2. ***KeyGen***$(PK, MSK, S) \to SK$*: on input the public parameters PK, the master secret key MSK and an attribute set S, output a private key SK associated with the set S. We assume that SK contains a description of the attribute set S.*
3. ***OffEnc***$(PK, \mathscr{M}) \to IntC$*: on input the public parameters PK and a message $\mathscr{M}$ from the message space $\mathscr{MS}$, output an intermediate ciphertext IntC.*

4. ***ReKeyGen***$(PK, SK, \mathbb{AS}) \rightarrow RK$*: on input the public parameters PK, a private key SK for the attribute set S and the access structure* $\mathbb{AS}$*, output a re-encryption key RK associated with the set S.*

5. ***OnEnc***$(PK, RK, IntC) \rightarrow CT$*: on input the public parameters PK, the re-encryption key RK for the attribute set S and associated with the access structure* $\mathbb{AS}$ *the intermediate ciphertext IntC, output a ciphertext* CT*.*

6. ***Dec***$(PK, SK', CT) \rightarrow \mathscr{M}$*: on input the public parameters PK, a private key* SK' *for the attribute set* S' *and a ciphertext CT associated with the access structure* $\mathbb{AS}$*, output a message* $\mathscr{M}$ *if* S' *satisfies* $\mathbb{AS}$*, or a symbol* $\perp$ *indicating either CT is invalid or* S' *does not satisfy* $\mathbb{AS}$*.*

The algorithms **Setup** and **KeyGen** are conducted by a trusted third party. The algorithm **OffEnc** can be executed by any user or any third party. The algorithm **ReKeyGen** is run by a user who has knowledge of the corresponding private key SK. The algorithm **OnEnc** can be run by the cloud. The algorithm **Dec** is run by a user who has knowledge of the corresponding private key SK'. In this scenario, we assume that the cloud is trusted and it will execute the required re-encryption algorithm correctly. We do not want to over-complicate the scenario by dealing with an untrusted cloud, as this will make the description of the scheme becomes very complicated. We note that dealing with a non-trusted cloud can be handled using different mechanisms, which is outside the scope of this work.

Correctness

For an attribute universe $\mathscr{U}$ and a security parameter $\lambda \in \mathbb{N}$, for all $(PK, MSK) \leftarrow$ **Setup**$(\lambda, \mathscr{U})$, for all $S, S' \subseteq \mathscr{U}$, for all access structure $\mathbb{AS} \in \mathscr{AS}$, for all message $\mathscr{M} \in \mathscr{MS}$, if $SK \leftarrow$ **KeyGen**(PK, MSK, S), $RK \leftarrow$ **ReKeyGen**$(PK, SK, \mathbb{AS})$, $SK' \leftarrow$ **KeyGen**(PK, MSK, S') and $S' \in \mathbb{AS}$, then $\mathscr{M} \leftarrow$ **Dec**$(PK, SK',$ **OnEnc**$(PK, RK,$ **OffEnc**$(PK, \mathscr{M}))$.

The algorithms in OO-CP-AB-PRE are depicted in Fig. 2.
In our scheme, we require that only a valid decryptor of the intermediate ciphertext *IntC* can generate a valid re-encryption key. Moreover, an attacker can generate any re-encryption key it wishes given an intermediate ciphertext *IntC*, but the final ciphertext will be invalid. This means that nobody can in fact decrypt the ciphertext correctly and retrieve the plaintext. More precisely, consider the following attack. The attacker is given *IntC* that depends only on the public key *PK* and the message $\mathscr{M}$, such that it has insufficient attribute set S'. It can generate the re-encryption key RK' by running the **ReKeyGen** algorithm with input its own private key SK' and access structure (M, ρ) that it selects (meaning that this access structure allows the attacker to decrypt the final ciphertexts). It then proceeds to the online encryption of *IntC* using the previous re-encryption RK' that it created. It obtains a final ciphertext CT. Thus, the attacker can decrypt CT running the **Decrypt** algorithm with input its own private key SK'. We call this type of attacks as a "trivial attack". We exclude this trivial attack from our security model.

3.2 Security Models

In this section, we first describe the security against outsider attacks and then, we focus on the security against insider attacks.

3.2.1 Outsider Attacks

From an outsider's point of view, we consider outsider attacks, which deals with selective IND-CCA security, as outlined below.

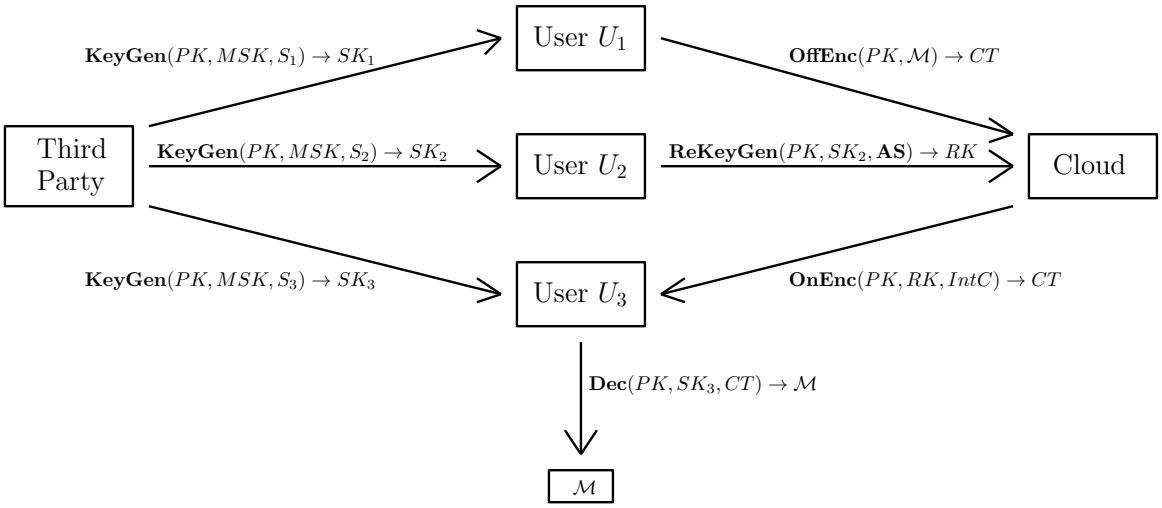

Figure 2: A third party generates the private keys for users U_1, U_2 and U_3. U_1 generates the intermediate ciphertext *IntC* and sends it to the cloud. U_2 generates the re-encryption key *RK* and sends it to the cloud. The cloud generates the final ciphertext *CT* and sends it to user U_3. U_3 successfully decrypts *CT* if the attribute set S_3 related to the private key SK_3 satisfies $\mathbb{AS}$.

Selective IND-CCA Security An OO-CP-ABE-PRE scheme scheme is selectively IND-CCA secure if no probabilistic polynomial-time (PPT) adversary $\mathscr{A}$ can win the game below with non-negligible advantage. In the game, $\mathscr{B}$ is the simulator, λ is the security parameter, and $\mathscr{U}$ is the attribute universe.

Init. $\mathscr{A}$ outputs a challenge access structure (M^*, ρ^*) to $\mathscr{B}$.

Setup. $\mathscr{B}$ runs the **Setup** algorithm and gives the public parameters *PK* to $\mathscr{A}$.

Phase 1. $\mathscr{A}$ is given access to the following oracles:

1. *Private key extraction oracle* $O_{sk}(S)$*:* on input an attribute set S, $\mathscr{B}$ runs $SK \leftarrow \mathbf{KeyGen}(PK, MSK, S)$ and returns SK to $\mathscr{A}$.

2. *Re-encryption key extraction oracle* $O_{rk}(S, (M, \rho))$*:* on input an attribute set S and an access structure (M, ρ), $\mathscr{B}$ runs $RK \leftarrow \mathbf{ReKeyGen}(PK, SK, (M, \rho))$ and returns RK to $\mathscr{A}$, where $SK \leftarrow \mathbf{KeyGen}(PK, MSK, S)$.

3. *Online encryption oracle* $O_{one}(S, (M, \rho), IntC)$*:* on input an attribute set S, an access structure (M, ρ) and an intermediate ciphertext *IntC*, $\mathscr{B}$ runs $CT \leftarrow \mathbf{OnEnc}(PK, RK, IntC)$ and returns CT to $\mathscr{A}$, where $RK \leftarrow \mathbf{ReKeyGen}(PK, SK, (M, \rho))$ and $SK \leftarrow \mathbf{KeyGen}(PK, MSK, S)$.

4. *Ciphertext decryption oracle* $O_d(S, CT)$*:* on input an attribute set S and a ciphertext CT, $\mathscr{B}$ runs $\mathscr{M} \leftarrow \mathbf{Dec}(PK, SK, CT)$ and returns $\mathscr{M}$ to $\mathscr{A}$, where $SK \leftarrow \mathbf{KeyGen}(PK, MSK, S)$ and S satisfies (M, ρ).

We notice that if the ciphertexts queried to oracles O_{one} and O_d are invalid, then $\mathscr{B}$ simply outputs $\perp$. In this phase, it is forbidden to issue the following queries:

- $O_{sk}(S)$ for any S satisfying (M^*, ρ^*),

- $O_{rk}(S,(M,\rho))$ for any S satisfying (M^*,ρ^*).

Challenge. $\mathscr{A}$ submits two equal length messages $\mathscr{M}_0$ and $\mathscr{M}_1$. $\mathscr{B}$ flips a random coin $b \in \{0,1\}$ and encrypts $\mathscr{M}_b$ online under (M^*,ρ^*). The ciphertext $CT^* \leftarrow \textbf{OnEnc}(PK,RK,IntC)$ is given to $\mathscr{A}$, such that $RK \leftarrow \textbf{ReKeyGen}(PK,SK,(M^*,\rho^*))$ and $IntC \leftarrow \textbf{OffEnc}(PK,\mathscr{M}_b)$.

Phase 2. Phase 1 is repeated except for the following:

1. $O_{sk}(S)$ for any S satisfying (M^*,ρ^*),
2. $O_{rk}(S,(M,\rho))$ for any S satisfying (M^*,ρ^*),
3. $O_{one}(S,(M,\rho),IntC)$ for any invalid intermediate ciphertext or invalid ciphertext where S satisfies (M^*,ρ^*),
4. $O_d(S,CT)$ for any invalid ciphertext or $CT = CT^*$ where S satisfies (M^*,ρ^*).

Guess. $\mathscr{A}$ outputs a guess $b' \in \{0,1\}$. If $b' = b$ then $\mathscr{A}$ wins.

The advantage of $\mathscr{A}$ is defined as $\varepsilon_1 = Adv^{IND-CCA}_{OO-CP-AB-PRE,\mathscr{A}}(\lambda,\mathscr{U}) = |Pr[b' = b] - 1/2|$.

Definition 5. *An OO-CP-AB-PRE is selectively IND-CCA secure for attribute universe $\mathscr{U}$ if no probabilistic polynomial-time (PPT) adversary $\mathscr{A}$ can win the game above with non-negligible advantage, meaning that $\varepsilon_1 \leq \frac{1}{2} + negl(\lambda)$, where negl is a negligible function.*

We illustrate the selective CCA security model in Fig 3.

3.2.2 Insider Attacks

Selective Collusion Resistance From an insider's point of view, we consider insider attacks, which deals with selective collusion resistance, as outlined below. The collusion resistance is also known as the master key security in the litterature [8, 20, 21, 22].

In the following security model, we omit the descriptions of the Online encryption and Ciphertext decryption oracles since they are straightforward: any re-encryption key could be correctly generated from the Re-encryption key extraction oracle.

An OO-CP-ABE-PRE scheme scheme is selectively collusion resistant if no probabilistic polynomial-time (PPT) adversary $\mathscr{A}$ can win the game below with non-negligible advantage. In the game, $\mathscr{B}$ is the simulator, λ is the security parameter, and $\mathscr{U}$ is the attribute universe.

Init. $\mathscr{A}$ outputs an attribute set S^* to $\mathscr{B}$.

Setup. $\mathscr{B}$ runs the **Setup** algorithm and gives the public parameters PK to $\mathscr{A}$.

Query Phase. $\mathscr{A}$ is given access to the following oracles:

1. *Private key extraction oracle $O_{sk}(S)$:* on input an attribute set $S \neq S^*$, $\mathscr{B}$ runs $SK \leftarrow \textbf{KeyGen}(PK,MSK,S)$ and returns SK to $\mathscr{A}$.
2. *Re-encryption key extraction oracle $O_{rk}(S,(M,\rho))$:* on input an attribute set S and an acces structure (M,ρ), $\mathscr{B}$ runs $RK \leftarrow \textbf{ReKeyGen}(PK,SK,(M,\rho))$ and returns RK to $\mathscr{A}$, where $SK \leftarrow \textbf{KeyGen}(PK,MSK,S)$.

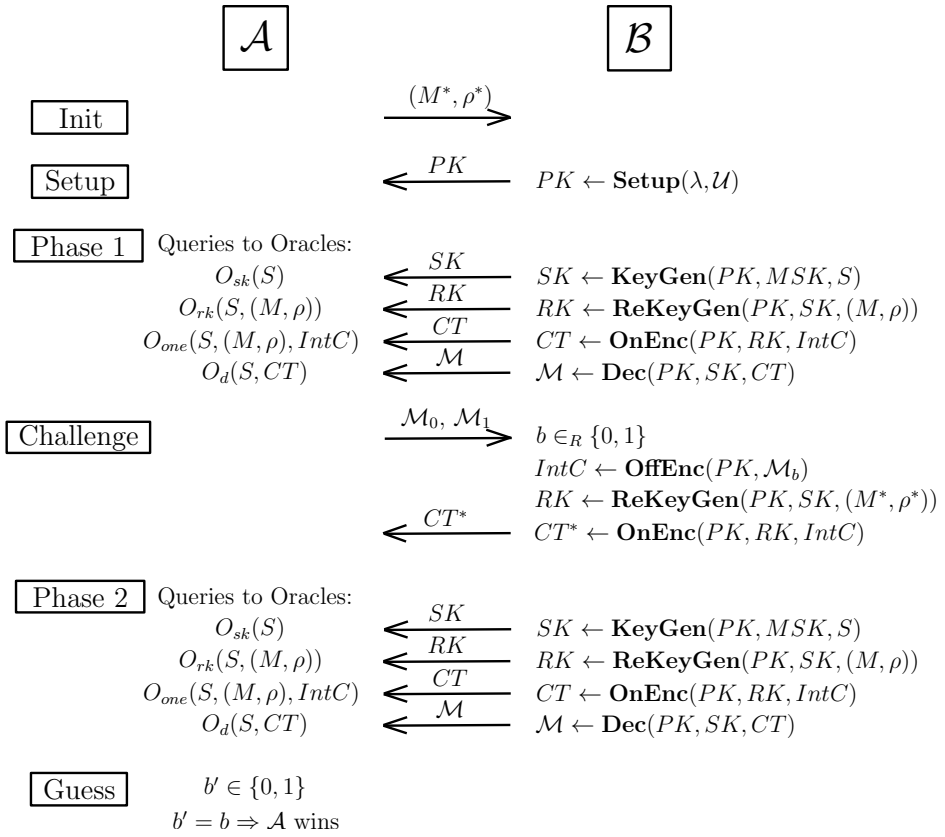

Figure 3: Selective CCA security Game between the adversary $\mathscr{A}$ and the simulator $\mathscr{B}$.

Output. $\mathscr{A}$ submits a private key SK^* for the S^*.

The advantage of $\mathscr{A}$ is defined as $\varepsilon_2 = Adv^{CR}_{OO-CP-AB-PRE,\mathscr{A}}(\lambda, \mathscr{U}) = Pr[\mathscr{A}$ succeeds$]$.

Definition 6. *An OO-CP-AB-PRE is selectively collusion resistant for attribute universe $\mathscr{U}$ if no probabilistic polynomial-time (PPT) adversary $\mathscr{A}$ can win the game above with non-negligible advantage, meaning that $\varepsilon_2 \leq negl(\lambda)$, where negl is a negligible function.*

We illustrate the selective collusion resistance model in Fig 4.

4 Online/Offline Ciphertext Attribute-based Proxy Re-Encryption (OO-CP-AB-PRE) Construction

Our scheme is based on the Online/Offline CP-ABE system developped by Hohenberger and Waters [15] and the CP-AB-PRE system proposed by Liang et al. [20]. One can now create an intermediate ciphertext in the offline phase and then can translate it to a ciphertext for a hitherto unknown access structure, under a re-encryption key generated from the sender's private key. The Hohenberger-Waters scheme [15] is based on the unbounded CP-ABE construction implemented by Rouselakis and Waters [28]. We assume the existence of a bound P on the maximum number of rows in an LSSS access structure that will be used to encrypt in the system below.

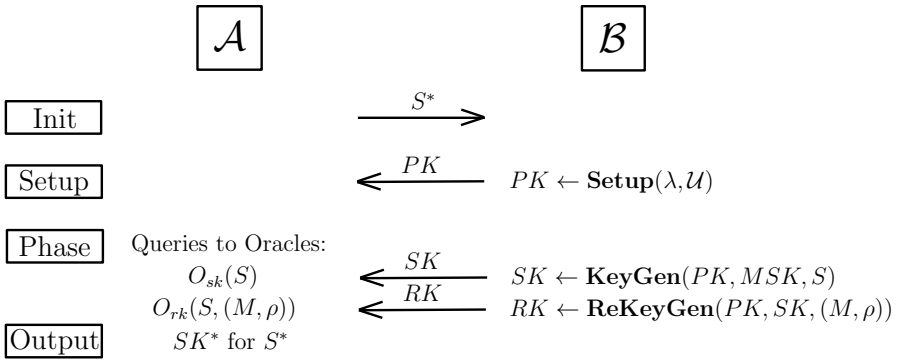

Figure 4: Selective collusion resistance Game between the adversary $\mathscr{A}$ and the simulator $\mathscr{B}$.

The idea is based on the Boneh-Boyen IBE system [6]. During the offline phase, a ciphertext is generated by encrypting to an exponent $x \in \mathbb{Z}_p$ chosen at random with another random element $s \in \mathbb{Z}_p$. This ciphertext has elements $C_1 = g^s$ and $C_2 = (u^x h)^s$. Moreover, the public parameters are a description of the bilinear group $\mathbb{G}$ of order p, along with the tupke $(g,u,h,e(g,g)^\alpha)$, and the encapsulated key is $K = e(g,g)^{\alpha s}$. The intermediate ciphertext $IntC = (C_1, C_2, x, s)$ is stored by the offline algorithm. During the online phase, given an identity $id \in \mathbb{Z}_p$, the encryptor adds a "correction factor" equal to $s \cdot (id - x) \in \mathbb{Z}_p$ to the components C_1 and C_2.

4.1 Construction

Setup$(\lambda, \mathscr{U})$. The **Setup** algorithm takes as input the security parameter λ and the attribute universe U viewed as $\mathbb{Z}_p$, and chooses two bilinear cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order $p \in \Theta(2^\lambda)$, along with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. It also chooses random generators $g, h, u, v, w \in \mathbb{G}$ and picks at random an exponent $\alpha \in \mathbb{Z}_p$. It defines five hash functions $H_1 : \mathbb{G}_T \to \{0,1\}^{2\lambda}$, $H_2 : \{0,1\}^{2\lambda} \to \mathbb{Z}_p$, $H_3 : \{0,1\}^\lambda \to \mathbb{Z}_p$, $H_4 : \{0,1\}^* \to \mathbb{G}$ and $H_5 : \{0,1\}^* \to \mathbb{G}$. It then sets the public parameters $PK = (p, \mathbb{G}, \mathbb{G}_T, e, g, h, u, v, w, e(g,g)^\alpha, H_1, H_2, H_3, H_4, H_5)$ and the master secret key $MSK = \alpha$.

KeyGen(PK, MSK, S). The **KeyGen** algorithm takes as input the public parameters PK, the master secret key MSK, and an attribute set $S = \{A_1, A_2, \cdots, A_k\} \subseteq \mathbb{Z}_p$. It chooses random values $r, r_1, r_2, \cdots, r_k \in \mathbb{Z}_p$. It then computes $K_0 = g^\alpha w^r$, $K_1 = g^r$ and for $i = 1, \cdots, k$, $K_{i,2} = g^{r_i}$ and $K_{i,3} = (u^{A_i} h)^{r_i} v^{-r}$. The private key is $SK = (S, K_0, K_1, \{K_{i,2}, K_{i,3}\}_{i \in [1,k]})$.

OffEnc$(PK, \mathscr{M})$. The **OffEnc** algorithm takes as input the public parameters PK and a message $\mathscr{M}$ from the message space $\{0,1\}^\lambda$. We assume there exists a maximum bound of P rows in any LSSS access structure used in a ciphertext. It first picks at random $\beta \in \{0,1\}^\lambda$ and computes $s = H_2(\beta, \mathscr{M})$, $B_1 = (\mathscr{M} \| \beta) \oplus H_1(e(g,g)^{\alpha s})$, $C_{0,1} = g^s$ and $C_{0,2} = h^s$. Then, for $j = 1, \cdots, P$, it chooses at random $\lambda'_j, t_j, x_j \in \mathbb{Z}_p$ and computes $C_{j,1} = w^{\lambda'_j} v^{t_j}$, $C_{j,2} = (u^{x_j} h)^{-t_j}$ and $C_{j,3} = g^{t_j}$. This process can be seen as encrypting for a random attribute x_j with a random "share" λ'_j of s, and it will be corrected in the online phase with the attributes defined in the **ReKeyGen** process. We notice that the work done in this offline phase is almost equivalent to the one of the regular encryption algorithm in [28]. Moreover, it computes $D = H_4(B_1, C_{0,1}, C_{0,2}, (C_{1,1}, C_{1,2}, C_{1,3}), \cdots, (C_{P,1}, C_{P,2}, C_{P,3}))^s$. Eventually, the intermediate ciphertext is $IntC = (s, B_1, C_{0,1}, C_{0,2}, \{\lambda'_j, t_j, x_j, C_{j,1}, C_{2,j}, C_{3,j}\}_{j \in [1,P]}, D)$.

ReKeyGen$(PK, SK, (M, \rho))$. The **ReKeyGen** algorithm takes as input the public parameters PK,

the private key SK for attribute set $S = \{A_1, A_2, \cdots, A_k\} \subseteq \mathbb{Z}_p$, and a LSSS access structure (M, ρ), where M is a $l \times n$ matrix and $l \leq P$.

First, it first picks at random $\tilde{\beta}, \delta \in \{0,1\}^\lambda$ and computes $\tilde{s} = H_2(\tilde{\beta}, \delta)$, $\tilde{B}_1 = (\delta \| \tilde{\beta}) \oplus H_1(e(g,g)^{\alpha\tilde{s}})$, and $\tilde{C}_0 = g^{\tilde{s}}$. It picks at random $\tilde{y}_2, \cdots, \tilde{y}_n \in \mathbb{Z}_p$, sets the vector $\vec{\tilde{y}} = (\tilde{s}, \tilde{y}_2, \cdots, \tilde{y}_n)^T$, and computes a vector of shares of $\tilde{s}$ as $(\tilde{\lambda}_1, \cdots, \tilde{\lambda}_l)^T = M \cdot \vec{\tilde{y}}$. Then, for $j = 1, \cdots, l$, it chooses at random $\tilde{t}_j \in \mathbb{Z}_p$ and computes $\tilde{C}_{j,1} = w^{\tilde{\lambda}_j} v^{\tilde{t}_j}$, $\tilde{C}_{j,2} = (u^{\rho(j)} h)^{-\tilde{t}_j}$ and $\tilde{C}_{j,3} = g^{\tilde{t}_j}$. Moreover, it computes $\tilde{D} = H_5(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho))^{\tilde{s}}$. Eventually, the intermediate ciphertext for the re-encryption key is $Int\tilde{C} = ((M, \rho), \tilde{B}_1, \tilde{C}_0, \{\tilde{C}_{j,1}, \tilde{C}_{j,2}, \tilde{C}_{j,3}\}_{j \in [1,l]}, \tilde{D})$.

Second, it picks at random an exponent $\theta \in \mathbb{Z}_p$ and computes $RK_{0,1} = K_0^{H_3(\delta)} h^\theta$, $RK_{0,2} = g^\theta$, $RK_1 = K_1^{H_3(\delta)}$ and for $i = 1, \cdots, k$, $RK_{i,2} = K_{i,2}^{H_3(\delta)}$ and $RK_{i,3} = K_{i,3}^{H_3(\delta)}$.

The re-encryption key is $RK = (S, RK_{0,1}, RK_{0,2}, RK_1, \{RK_{i,2}, RK_{i,3}\}_{i \in [1,k]}, Int\tilde{C})$.

OnEnc$(PK, RK, IntC)$. The **OnEnc** algorithm takes as input the public parameters PK, the re-encryption key RK for attribute set S and associated with the access structure (M, ρ), and an intermediate ciphertext $IntC$.

First, it checks the validity of the intermediate ciphertext $IntC$ as follows:

$$
\begin{aligned}
& e(C_{0,1}, h) \stackrel{?}{=} e(g, C_{0,2}) \\
& \forall j \in [1,P], e(C_{j,1}, g) \stackrel{?}{=} e(w,g)^{\lambda'_j} \cdot e(v, C_{j,3}) \\
& \forall j \in [1,P], e(C_{j,2}, g) \stackrel{?}{=} e(u, C_{j,3}^{-1})^{x_j} \cdot e(h, C_{j,3}^{-1}) \\
& e(C_{0,1}, H_4(B_1, C_{0,1}, C_{0,2}, (C_{1,1}, C_{1,2}, C_{1,3}), \cdots, (C_{P,1}, C_{P,2}, C_{P,3}))) \stackrel{?}{=} e(g, D)
\end{aligned} \tag{1}
$$

and the validity of the re-encryption key RK as follows:

$$
e(\tilde{C}_0, H_5(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho))) \stackrel{?}{=} e(g, \tilde{D}).
$$

It takes the element s from $IntC$ and picks at random $y_2, \cdots, y_n \in \mathbb{Z}_p$, sets the vector $\vec{y} = (s, y_2, \cdots, y_n)^T$, where T denotes the transpose of the matrix, and computes a vector of shares of s as $(\lambda_1, \cdots, \lambda_l)^T = M \cdot \vec{y}$. Let $I \subset \{1, \cdots, l\}$ be defined as $I = \{i : \rho(i) \in S\}$, and $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ be a set of constants such that $\sum_{i \in I} w_i \cdot \lambda_i = s$. For $j = 1, \cdots, l$, it then computes $C_{j,4} = \lambda_j - \lambda'_j$ and $C_{j,5} = t_j \cdot (\rho(j) - x_j)$. Intuitively, this corrects to the proper attributes and shares of s. Finally, it computes the value B_2 as the quotient:

$$
B_2 = \frac{e(C_{0,1}, RK_{0,1}) / e(C_{0,2}, RK_{0,2})}{e(w^{\sum_{i \in I} C_{i,4} \cdot w_i}, RK_1) \cdot \prod_{i \in I} (e(C_{i,1}, RK_1) \cdot e(C_{i,2} \cdot u^{C_{i,5}}, RK_{j,2}) \cdot e(C_{i,3}, RK_{j,3}))^{w_i}},
$$

where j is the index of the attribute $\rho(i)$ in S (it depends on i).

It sets the ciphertext as $CT = ((M, \rho), C_{0,1}, B_2, Int\tilde{C})$.

Dec(PK, SK, CT). The **Dec** algorithm recovers the message $\mathscr{M}$. It takes as input the public parameters PK, the ciphertext $CT = ((M, \rho), C_{0,1}, B_2, Int\tilde{C})$ for access structure (M, ρ), and a private key $SK = (S, K_0, K_1, \{K_{i,2}, K_{i,3}\}_{i \in [1,k]})$ for attribute set S. If S does not satisfy this access structure (M, ρ), then the algorithm issues $\perp$. Otherwise, it sets $I \subset \{1, \cdots, l\}$ as $I = \{i : \rho(i) \in S\}$ and finds the set of constants $\{\tilde{w}_i \in \mathbb{Z}_p\}_{i \in I}$ such that $\sum_{i \in I} \tilde{w}_i \cdot \tilde{\lambda}_i = \tilde{s}$ (let M_i be the i-th row pf the matrix M, we have that $\sum_{i \in I} \tilde{w}_i \cdot M_i = (1, 0, \cdots, 0)$). Then, it checks that:

$$
\begin{aligned}
& S \text{ satisifies } (M, \rho)? \\
& e(\tilde{C}_0, H_5(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho))) \stackrel{?}{=} e(g, \tilde{D})
\end{aligned} \tag{2}
$$

If the above equations hold, then it proceeds; otherwise, it outputs $\perp$.

To recover the message $\mathscr{M}$, the algorithm first recovers the value $e(g,g)^{\alpha\tilde{s}}$ by calculating:

$$\frac{e(\tilde{C}_0, K_0)}{\prod_{i\in I}(e(\tilde{C}_{i,1}, K_1)\cdot e(\tilde{C}_{i,2}, K_{j,2})\cdot e(\tilde{C}_{i,3}, K_{j,3}))^{\tilde{w}_i}}$$

where j is the index of the attribute $\rho(i)$ in S (it depends on i). Afterwards, it computes $H_1(e(g,g)^{\alpha\tilde{s}}) \oplus \tilde{B}_1 = H_1(e(g,g)^{\alpha\tilde{s}}) \oplus (\delta \| \tilde{\beta}) \oplus H_1(e(g,g)^{\alpha\tilde{s}}) = \delta \| \tilde{\beta}$. If $\tilde{C}_0 = g^{H_2(\tilde{\beta},\delta)}$, then it proceeds; otherwise, it outputs $\perp$. Finally, it computes $H_1(B_2^{1/H_3(\delta)}) \oplus B_1 = H_1(e(g,g)^{\alpha\cdot s}) \oplus \mathscr{M} \| \beta \oplus H_1(e(g,g)^{\alpha\cdot s}) = \mathscr{M} \| \beta$, and outputs $\mathscr{M}$ if $C_{0,1} = g^{H_2(\beta,\mathscr{M})}$; otherwise it outputs $\perp$.

Correctness

We show the correctness of the above construction as follows. If the attribute set S of the ciphertext is authorized, then $\sum_{i\in I} w_i \cdot \lambda_i = s$ and $\sum_{i\in I} \tilde{w}_i \cdot \tilde{\lambda}_i = \tilde{s}$. We recall that $\rho(i) = A_j$. Therefore, the value B_2 is recovered as follows:

$$\begin{aligned}
B_2 &= \frac{e(C_{0,1}, RK_{0,1})/e(C_{0,2}, RK_{0,2})}{e(w^{\sum_{i\in I} C_{i,4}\cdot w_i}, RK_1)\cdot \prod_{i\in I}(e(C_{i,1}, RK_1)\cdot e(C_{i,2}\cdot u^{-C_{i,5}}, RK_{j,2})\cdot e(C_{i,3}, RK_{j,3}))^{w_i}} \\
&= \frac{e(C_{0,1}, K_0^{H_3(\delta)} h^{\theta})/e(C_{0,2}, g^{\theta})}{e(w^{\sum_{i\in I} C_{i,4}\cdot w_i}, K_1^{H_3(\delta)})\cdot \prod_{i\in I}(e(C_{i,1}, K_1^{H_3(\delta)})\cdot e(C_{i,2}\cdot u^{-C_{i,5}}, K_{j,2}^{H_3(\delta)})\cdot e(C_{i,3}, K_{j,3}^{H_3(\delta)}))^{w_i}} \\
&= \frac{(e(g^s, g^{\alpha} w^r)^{H_3(\delta)}\cdot e(g^s, h^{\theta}))/e(h^s, g^{\theta})}{e(w^{\sum_{i\in I}(\lambda_i - \lambda_i')w_i}, g^r)^{H_3(\delta)}\cdot \prod_{i\in I}(e(w^{\lambda_i'} v^{t_i}, g^r)\cdot e((u^{x_i}h)^{-t_i}\cdot u^{-t_i(\rho(i)-x_i)}, g^{r_j})\cdot e(g^{t_i}, (u^{A_j}h)^{r_j} v^{-r}))^{H_3(\delta)\cdot w_i}} \\
&= \frac{e(g,g)^{\alpha s\cdot H_3(\delta)}\cdot e(g,w)^{sr\cdot H_3(\delta)}}{(e(g,w)^{\sum_{i\in I}(\lambda_i-\lambda_i')r}\cdot \prod_{i\in I} e(g,w)^{\lambda_i' r} e(g,v)^{t_i r} e(g,u)^{-\rho(i) t_i r_j} e(g,h)^{-t_i r_j} e(g,u)^{A_j t_i r_j} e(g,h)^{t_i r_j} e(g,v)^{-t_i r})^{H_3(\delta)\cdot w_i}} \\
&= \frac{e(g,g)^{\alpha s\cdot H_3(\delta)}\cdot e(g,w)^{sr\cdot H_3(\delta)}}{(e(g,w)^{\sum_{i\in I}(\lambda_i-\lambda_i')r}\cdot \prod_{i\in I} e(g,w)^{\lambda_i' r})^{H_3(\delta)\cdot w_i}} \\
&= \frac{e(g,g)^{\alpha s\cdot H_3(\delta)}\cdot e(g,w)^{sr\cdot H_3(\delta)}}{e(g,w)^{r\cdot H_3(\delta)\sum_{i\in I}\lambda_i w_i}} = e(g,g)^{\alpha s\cdot H_3(\delta)}
\end{aligned}$$

and the value $e(g,g)^{\alpha\tilde{s}}$ is recovered as follows:

$$\begin{aligned}
&\frac{e(\tilde{C}_0, K_0)}{\prod_{i\in I}(e(\tilde{C}_{i,1}, K_1)\cdot e(\tilde{C}_{i,2}, K_{j,2})\cdot e(\tilde{C}_{i,3}, K_{j,3}))^{\tilde{w}_i}} \\
= &\frac{e(g^{\tilde{s}}, g^{\alpha} w^r)}{\prod_{i\in I}(e(w^{\tilde{\lambda}_i} v^{\tilde{t}_i}, g^r)\cdot e((u^{\rho(i)}h)^{-\tilde{t}_i}, g^{r_j})\cdot e(g^{\tilde{t}_i}, (u^{A_j}h)^{r_j} v^{-r}))^{\tilde{w}_i}} \\
= &\frac{e(g,g)^{\alpha\tilde{s}}\cdot e(g,w)^{\tilde{s}r}}{(\prod_{i\in I} e(g,w)^{\tilde{\lambda}_i r} e(g,v)^{\tilde{t}_i r} e(g,u)^{-\rho(i)\tilde{t}_i r_j} e(g,h)^{-\tilde{t}_i r_j} e(g,u)^{A_j \tilde{t}_i r_j} e(g,h)^{\tilde{t}_i r_j} e(g,v)^{-\tilde{t}_i r})^{\tilde{w}_i}} \\
= &\frac{e(g,g)^{\alpha\tilde{s}}\cdot e(g,w)^{\tilde{s}r}}{\prod_{i\in I} e(g,w)^{\tilde{\lambda}_i r\tilde{w}_i}} \\
= &\frac{e(g,g)^{\alpha\tilde{s}}\cdot e(g,w)^{\tilde{s}r}}{e(g,w)^{r\sum_{i\in I}\tilde{\lambda}_i\tilde{w}_i}} = e(g,g)^{\alpha\tilde{s}}
\end{aligned}$$

5 Security Proof

5.1 Assumptions

We first recall the definitions of the Computational BDHE and Decisional q-parallel Bilinear Diffie-Hellman Exponent assumption, and subsequently, we recall the assumption similar to the previous one, called the $q-1$ assumption as defined in [28].

Given a security parameter $\lambda \in \mathbb{N}$, we assume that there is a group generator algorithm $\mathbf{GroupGen}(\lambda) \rightarrow (p, \mathbb{G}, \mathbb{G}_T, e)$ that outputs the description of the symmetric bilinear group of order $p = \Theta(2^\lambda)$.

Definition 7 (Computational BDHE Assumption)**.** *Given an integer q polynomial in λ, a generator g of $\mathbb{G}$ and a tuple $T =$*

$$g, g^a g^b, g^c$$

the computational BDHE problem is to output $Z = e(g,g)^{abc}$, where $a, b, c \in_R \mathbb{Z}_p$. We define $Succ_{\mathscr{A}}^{CBDHE} = Pr[\mathscr{A}(T, e(g,g)^{abc}) = 0]$ as the success of an adversary $\mathscr{A}$ in solving the computational BDHE problem. We say that the computational BDHE assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ if no PPT algorithm has non-negligible success.

Definition 8 (Decisional q-parallel BDHE Assumption)**.** *Given an integer q polynomial in λ, a generator g of $\mathbb{G}$ and a tuple $T =$*

$$\begin{aligned}
&g, g^s, g^a, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}} \\
&\forall 1 \leq j \leq q, g^{s \cdot b_j}, g^{a/b_j}, \cdots, g^{a^q/b_j}, g^{a^{q+2}/b_j}, \cdots, g^{a^{2q}/b_j} \\
&\forall 1 \leq j, k \leq q, j \neq k, g^{a \cdot s \cdot b_k / b_j}, \cdots, g^{a^q \cdot s \cdot b_k / b_j}
\end{aligned}$$

the decisional q-parallel BDHE problem is to decide whether $Z = e(g,g)^{a^{q+1} \cdot s}$, where $a, s, b_1, \cdots, b_q \in_R \mathbb{Z}_p$, or $Z = R \in_R \mathbb{G}_T$. We define $Adv_{\mathscr{A}}^{D-q-PBDHE} = |Pr[\mathscr{A}(T, e(g,g)^{a^{q+1} \cdot s}) = 0] - Pr[\mathscr{A}(T,R) = 0]|$ as the advantage of an adversary $\mathscr{A}$ 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.

Definition 9 ($q-1$ Assumption)**.** *Given an integer q polynomial in λ, a generator g of $\mathbb{G}$ and a tuple $T =$*

$$\begin{aligned}
&g, g^s \\
&\forall 1 \leq i, j \leq q, g^{a^i}, g^{b_j}, g^{s \cdot b_j}, g^{a^i \cdot b_j}, g^{a^i / b_j^2} \\
&\forall 1 \leq i \leq 2q, 1 \leq j \leq q, i \neq q+1, g^{a^i / b_j} \\
&\forall 1 \leq i \leq 2q, 1 \leq j, k \leq q, j \neq k, g^{a^i b_j / b_k^2} \\
&\forall 1 \leq i \leq 2q, 1 \leq j, k \leq q, j \neq k, g^{s \cdot a^i b_j / b_k}, g^{s \cdot a^i b_j / b_k^2}
\end{aligned}$$

the $q-1$ problem is to decide whether $Z = e(g,g)^{a^{q+1} \cdot s}$, where $a, s, b_1, \cdots, b_q \in_R \mathbb{Z}_p$, or $Z = R \in_R \mathbb{G}_T$. We define $Adv_{\mathscr{A}}^{q-1} = |Pr[\mathscr{A}(T, e(g,g)^{a^{q+1} \cdot s}) = 0] - Pr[\mathscr{A}(T,R) = 0]|$ as the advantage of an adversary $\mathscr{A}$ in winning the $q-1$ problem. We say that the $q-1$ assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ if no PPT algorithm has non-negligible advantage.

The hardness of this assumption has been studied in [28].

5.2 Outsider Attacks

Selective IND-CCA Security Proof

Theorem 1. *Suppose the $q-1$ assumption holds in $(\mathbb{G},\mathbb{G}_T)$ and H_1, H_2, H_3, H_4 and H_5 are hash functions, our OO-CP-AB-PRE scheme is selectively IND-CCA secure in the random oracle model.*

Proof. Suppose there is an adversary $\mathscr{A}$ who can break the IND-CCA security of our scheme. We then construct a reduction algorithm $\mathscr{B}$ to decide whether Z is either equal to $e(g,g)^{a^{q+1}\cdot s}$ or to a random element R in $\mathbb{G}_T$. The simulator $\mathscr{B}$ plays the IND-CCA game with $\mathscr{A}$ as follows.

$\mathscr{B}$ takes as input $(p,\mathbb{G},\mathbb{G}_T,e)$, a $q-1$ problem instance T and Z, where Z is either equal to $e(g,g)^{a^{q+1}\cdot s}$ or to a random element R in $\mathbb{G}_T$.

Initialization. The adversary gives the challenge access structure (M^*,ρ^*) to $\mathscr{B}$, where M^* has l rows and n columns such that $l \leq P,q$ and $n \leq q$.

Setup. The simulator chooses at random $\alpha' \in_R \mathbb{Z}_p$ and implicitly sets $\alpha = \alpha' + a^{q+1}$ by letting $e(g,g)^{\alpha} = e(g^a, g^{a^q})e(g,g)^{\alpha'}$ (it can be seen that $\alpha = \alpha' + a^{q+1}$, which cannot be computed by $\mathscr{B}$). We note that α is correctly distributed. $\mathscr{B}$ also picks at random $\hat{v},\hat{u},\hat{h} \in_R \mathbb{Z}_p$ and gives the following public parameters to $\mathscr{A}$:

$$w = g^a \qquad v = g^{\hat{v}} \cdot \prod_{\substack{j=1,\cdots,l \\ k=1,\cdots,n}} \left(g^{a^k/b_j}\right)^{M^*_{j,k}}$$

$$u = g^{\hat{u}} \cdot \prod_{\substack{j=1,\cdots,l \\ k=1,\cdots,n}} \left(g^{a^k/b_j^2}\right)^{M^*_{j,k}} \qquad h = g^{\hat{h}} \cdot \prod_{\substack{j=1,\cdots,l \\ k=1,\cdots,n}} \left(g^{a^k/b_j^2}\right)^{-\rho^*(j)M^*_{j,k}}$$

Since a is information-theoretically hidden from $\mathscr{A}$, the value w is properly uniformly random in $\mathbb{G}$. The values v,u,h are properly distributed due to $\hat{v},\hat{u},\hat{h}$ respectively. We note that all these values are calculated by $\mathscr{B}$ using terms from the assumption instance and (M^*,ρ^*) given by $\mathscr{A}$.

Then the simulator chooses the hash functions as in the real scheme, and sends the public parameters $PK = (p,\mathbb{G},\mathbb{G}_T,e,g,h,,u,v,w,e(g,g)^{\alpha},H_1,H_2,H_3,H_4,H_5)$ to $\mathscr{A}$. We note that the public parameters are identical to those in the real scheme for the adversary. At any time, $\mathscr{A}$ can adaptively query the random oracles H_j for $j \in [1,5]$, which are controlled by $\mathscr{B}$. The simulator maintains the lists H_j^{List} for $j \in [1,5]$, which are initially empty, and answers the queries to the random oracles as follows.

- H_1: on receipt of an H_1 query on $R \in \mathbb{G}_T$, if there is a tuple $(R,\delta_1) \in H_1^{List}$, $\mathscr{B}$ forwards the predefined value δ_1 to $\mathscr{A}$, where $\delta_1 \in \{0,1\}^{2\lambda}$. Otherwise, $\mathscr{B}$ sets $H_1(R) = \delta_1$, responds δ_1 to $\mathscr{A}$ and adds the tuple (R,δ_1) to H_1^{List}, where $\delta_1 \in_R \{0,1\}^{2\lambda}$.

- H_2: on receipt of an H_2 query on $(\beta,\mathscr{M})$, if there is a tuple $(\beta,\mathscr{M},s) \in H_2^{List}$, $\mathscr{B}$ forwards the predefined value s to $\mathscr{A}$, where $s \in \mathbb{Z}_p$. Otherwise, $\mathscr{B}$ sets $H_2(\beta,\mathscr{M}) = s$, responds s to $\mathscr{A}$ and adds the tuple $(\beta,\mathscr{M},s)$ to H_2^{List}, where $s \in_R \mathbb{Z}_p$.

- H_3: on receipt of an H_3 query on $\delta \in \{0,1\}^{\lambda}$, if there is a tuple (δ,ξ_1) in H_3^{List}, $\mathscr{B}$ forwards the predefined value ξ_1 to $\mathscr{A}$, where $\xi_1 \in \mathbb{Z}_p$. Otherwise, $\mathscr{B}$ sets $H_3(\delta) = \xi_1$ to $\mathscr{A}$ and adds the tuple (δ,ξ_1) to H_3^{List}, where $\xi_1 \in_R \mathbb{Z}_p$.

- H_4: on receipt of an H_4 query on $(B_1,C_{0,1},C_{0,2},(C_{1,1},C_{1,2},C_{1,3}),\cdots,(C_{l,1},C_{l,2},C_{l,3}))$, if there is a tuple $(B_1,C_{0,1},C_{0,2},(C_{1,1},C_{1,2},C_{1,3}),\cdots,(C_{l,1},C_{l,2},C_{l,3}),\xi_2,\delta_2) \in H_4^{List}$, $\mathscr{B}$ forwards the predefined value δ_2 to $\mathscr{A}$, where $\xi_2 \in \mathbb{Z}_p, \delta_2 \in \mathbb{G}$. Otherwise, $\mathscr{B}$ sets $\delta_2 = g^{\xi_2}$, responds δ_2 to $\mathscr{A}$ and adds the tuple $(B_1,C_{0,1},C_{0,2},(C_{1,1},C_{1,2},C_{1,3}),\cdots,(C_{l,1},C_{l,2},C_{l,3}),\xi_2,\delta_2)$ in H_4^{List}, where $\xi_2 \in_R \mathbb{Z}_p$.

- H_5: on receipt of an H_5 query on $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho))$, if there is a tuple $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho), \xi_3, \delta_3) \in H_5^{List}$, $\mathscr{B}$ forwards the predefined value δ_2 to $\mathscr{A}$, where $\xi_3 \in \mathbb{Z}_p, \delta_3 \in \mathbb{G}$. Otherwise, $\mathscr{B}$ sets $\delta_3 = g^{\xi_3}$, responds δ_3 to $\mathscr{A}$ and adds the tuple $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho), \xi_3, \delta_3)$ in H_5^{List}, where $\xi_3 \in_R \mathbb{Z}_p$.

In addition, $\mathscr{B}$ maintains the lists SK^{List}, RK^{List} and ONE^{List} which are initially empty as follows:

- SK^{List} records the tuples (S, SK) which are the results of the queries to $O_{sk}(S)$.
- RK^{List} records the tuples $(S, (M, \rho), \delta, \tilde{\beta}, RK, tag_1, tag_2, tag_3)$ which are the results of the queries to $O_{rk}(S, (M, \rho))$, where tag_1, tag_2 and tag_3 denote that the re-encryption key is randomly chosen, generated in O_{one} or in O_{rk}, respectively.
- ONE^{List} records the tuples $(S, (M, \rho), CT, tag_1, tag_2, tag_3)$ which are the results of the queries to $O_{one}(S, (M, \rho),\ IntC)$, where tag_1, tag_2 and tag_3 denote that the ciphertext is generated under a valid re-encryption key, under a randomly chosen re-encryption key or generated without any re-encryption key, respectively.

Phase 1. The simulator answers to $\mathscr{A}$'s queries as follows.

- *Private key extraction oracle $O_{sk}(S)$:*

  $\mathscr{B}$ has to produce private keys for non-authorized sets of attributes requested by $\mathscr{A}$. $\mathscr{B}$ proceeds in creating a key for an attribute set $S = \{A_1, \cdots, A_{|S|}\}$ as follows.

  Since S does not satisfy (M^*, ρ^*), there exists a vector $\vec{w} = (w_1, \cdots, w_n)^T \in \mathbb{Z}_p^n$ such that $w_1 = -1$ and $M_i^* \cdot \vec{w} = 0$ for all $i \in I = \{i : \rho^*(i) \in S\} \subseteq \{1, \cdots, l\}$ (if S satisfies (M^*, ρ^*) then the simulator outputs a random bit in $\{0,1\}$ and aborts the simulation). $\mathscr{B}$ then picks at random $\hat{r} \in_R \mathbb{Z}_p$ and implicitly computes

$$r = \hat{r} + w_1 a^q + \cdots + w_n a^{q+1-n} = \hat{r} + \sum_{i=1,\cdots,n} w_i a^{q+1-i}.$$

  This is properly distributed due to $\hat{r}$. Then $\mathscr{B}$ computes

$$\begin{aligned} K_0 &= g^{\alpha} w^r = g^{a^{q+1}} g^{\alpha'} g^{a\hat{r}} \prod_{i=1,\cdots,n} g^{w_i a^{q+2-i}} = g^{\alpha'} (g^a)^{\hat{r}} \prod_{i=2,\cdots,n} (g^{a^{q+2-i}})^{w_i} \\ K_1 &= g^r = g^{\hat{r}} \prod_{i=1,\cdots,n} (g^{a^{q+1-i}})^{w_i} \end{aligned}$$

  Moreover, for all $j = 1, \cdots, |S|$, $\mathscr{B}$ calculates the values $K_{j,2} = g^{r_j}$ and $K_{j,3} = (u^{A_j} h)^{r_j} v^{-r}$ as follows

$$\begin{aligned} v^{-r} &= v^{\hat{r}} \cdot (g^{\hat{v}} \prod_{\substack{j'=1,\cdots,l \\ k=1,\cdots,n}} g^{a^k M^*_{j',k}/b_{j'}})^{-\sum_{i=1,\cdots,n} w_i a^{q+1-i}} \\ &= v^{\hat{r}} \cdot \prod_{i=1,\cdots,n} (g^{a^{q+1-i}})^{\hat{v} w_i} \cdot \prod_{\substack{i,k=1,\cdots,n \\ j'=1,\cdots,l}} g^{-w_i M^*_{j',k} a^{q+1+k-i}/b_{j'}} \\ &= v^{\hat{r}} \cdot \prod_{i=1,\cdots,n} (g^{a^{q+1-i}})^{\hat{v} w_i} \cdot \prod_{\substack{i,k=1,\cdots,n, i\neq k \\ j'=1,\cdots,l}} (g^{a^{q+1+k-i}/b_{j'}})^{-w_i M^*_{j',k}} \cdot \prod_{\substack{i=1,\cdots,n \\ j'=1,\cdots,l}} g^{-w_i M^*_{j',k} a^{q+1}/b_{j'}} \\ &= Q \cdot \prod_{j'=1,\cdots,l} g^{-\vec{w} \cdot M^*_{j'} a^{q+1}/b_{j'}} = Q \cdot \prod_{j'=1,\cdots,l, \rho(j) \notin S} g^{-\vec{w} \cdot M^*_{j'} a^{q+1}/b_{j'}} \end{aligned}$$

such that $Q = v^{\hat{r}} \prod_{i=1,\cdots,n} (g^{a^{q+1-i}})^{\hat{r}w_i} \cdot \prod_{\substack{i,k=1,\cdots,n,i\neq k \\ j'=1,\cdots,l}} (g^{a^{q+1+k-i}/b_{j'}})^{-w_i M^*_{j',k}}$. The value Q can be calculated by $\mathscr{B}$ using the assumption and $\prod_{j'=1,\cdots,l,\rho(j)\notin S} g^{-\vec{w}\cdot M^*_{j'} a^{q+1}/b_{j'}}$ should be wiped out by $(u^{A_j}h)^{r_j}$. Therefore, for all attribute $A_j \in S$, $\mathscr{B}$ implicitly computes

$$\begin{aligned} r_j &= \hat{r}_j + r \cdot \sum_{i'=1,\cdots,n,\rho^*(i')\notin S} \frac{b_{i'}}{A_j - \rho^*(i')} \\ &= \hat{r}_j + \hat{r} \cdot \sum_{i'=1,\cdots,n,\rho^*(i')\notin S} \frac{b_{i'}}{A_j - \rho^*(i')} + \sum_{\substack{i=1,\cdots,n \\ i'=1,\cdots,l,\rho^*(i')\notin S}} \frac{w_i b_{i'} a^{q+1-i}}{A_j - \rho^*(i')} \end{aligned}$$

such that $\hat{r}_j$ is randomly chosen in $\mathbb{Z}_p$ and thus, r_j is properly distributed. The values b_i at the numerator cancel out with the values b_i^2 at the denominator, and thus cancel out the unknown part of v^{-r}.

Moreover, we note that r_j is well defined only for the attributes in the unauthorized set S or the unrelated attributes (not in the policy), since the sum is over the i''s such that $\rho^*(i') \notin S$. Thus, for all $A_j \in S$ or all $A_j \notin \{\rho^*(1),\cdots,\rho^*(l)\}$, the denominators $A_j - \rho^*(i')$ are non-zero. If $\mathscr{B}$ tries to put more attributes in the policy, and possibly create a key for an unauthorized set, then it would have to divide by zero. Then, we obtain $K_{j,3} = (u^{A_j}h)^{r_j} v^{-r}$ such that

$$\begin{aligned} (u^{A_j}h)^{r_j} &= (u^{A_j}h)^{\hat{r}_j} \cdot \Big(g^{\hat{u}A_j+\hat{h}} \prod_{\substack{i=1,\cdots,l \\ k=1,\cdots,n}} g^{(A_j-\rho^*(i))M^*_{i,k}a^k/b_i^2}\Big)^{\hat{r}\cdot\sum_{i'=1,\cdots,l,\rho^*(i')\notin S} \frac{b_{i'}}{A_j-\rho^*(i')}} \\ &\quad \cdot \Big(g^{\hat{u}A_j+\hat{h}} \prod_{\substack{i=1,\cdots,l \\ k=1,\cdots,n}} g^{(A_j-\rho^*(i))M^*_{i,k}a^k/b_i^2}\Big)^{\sum_{\substack{i'=1,\cdots,l,\rho^*(i')\notin S \\ j'=1,\cdots,n}} \frac{w_{j'}b_{i'}a^{q+1-j'}}{A_j-\rho^*(i')}} \\ &= (u^{A_j}h)^{\hat{r}_j} \cdot (K_{j,2}/g^{\hat{r}_j})^{\hat{u}A_j+\hat{h}} \cdot \prod_{\substack{i,i'=1,\cdots,l,\rho^*(i')\notin S \\ k=1,\cdots,n}} g^{\hat{r}(A_j-\rho^*(i))M^*_{i,k}b_{i'}a^k/(A_j-\rho^*(i'))b_i^2} \\ &\quad \cdot \prod_{\substack{i,i'=1,\cdots,l,\rho^*(i')\notin S \\ j',k=1,\cdots,n}} g^{(A_j-\rho^*(i))w_{j'}M^*_{i,k}b_{i'}a^{q+1+k-j'}/(A_j-\rho^*(i'))b_i^2} \\ &= Q' \cdot \prod_{\substack{i=1,\cdots,l,\rho^*(i)\notin S \\ i'=1,\cdots,n}} g^{(A_j-\rho^*(i))w_{i'}M^*_{i,i'}b_i a^{q+1+i'-i'}/(A_j-\rho^*(i))b_i^2} = Q' \cdot \prod_{i=1,\cdots,l,\rho^*\notin S} g^{(\vec{w}\cdot M^*_i)a^{q+1}/b_i} \end{aligned}$$

such that

$$\begin{aligned} Q' &= (u^{A_j}h)^{\hat{r}_j} \cdot (K_{j,2}/g^{\hat{r}_j})^{\hat{u}A_j+\hat{h}} \cdot \prod_{\substack{i,i'=1,\cdots,l,\rho^*(i')\notin S \\ k=1,\cdots,n}} (g^{b_{i'}a^k/b_i^2})^{\hat{r}(A_j-\rho^*(i))M^*_{i,k}/(A_j-\rho^*(i'))} \\ &\quad \cdot \prod_{\substack{i,i'=1,\cdots,l \\ j',k=1,\cdots,n \\ \rho^*(i')\notin S,(i\neq i'\vee j'\neq k)}} (g^{b_{i'}a^{q+1+k-j'}/b_i^2})^{(A_j-\rho^*(i))w_{j'}M^*_{i,k}/(A_j-\rho^*(i'))} \\ K_{j,2} &= g^{r_j} = g^{\hat{r}_j} \cdot \prod_{i'=1,\cdots,l,\rho^*(i')\notin S} (g^{b_{i'}})^{\hat{r}/(A_j-\rho^*(i'))} \cdot \prod_{\substack{i=1,\cdots,n \\ i'=1,\cdots,l,\rho^*(i')\notin S}} (g^{b_{i'}a^q+1-i})^{w_i/(A_j-\rho^*(i'))} \end{aligned}$$

The values Q' and $K_{j,2}$ can be calculated using the values from the problem instance. The second part of $(u^{A_j}h)^{r_j}$ cancels out with the unknown part of v^{-r}. Thus, $\mathscr{B}$ can compute $K_{j,2}$ and $K_{j,3}$ for all the attributes A_j in S and returns the private key $SK = (S, K_0, K_1, \{K_{j,2}, K_{j,3}\}_{j\in[1,|S|]})$ to $\mathscr{A}$.

- *Re-encryption key extraction oracle $O_{rk}(S,(M,\rho))$:*

  If $(S,(M,\rho),\delta,\tilde{\beta},RK,*,0,1) \in RK^{List}$, then the simulator returns RK to the adversary. Otherwise, $\mathscr{B}$ proceeds as follows.

  - If S satisfies (M^*,ρ^*) and $(S,SK) \in SK^{List}$, then $\mathscr{B}$ returns a random bit in $\{0,1\}$ and aborts the simulation.
  - If S satisfies (M^*,ρ^*) and $(S,SK) \notin SK^{List}$, then $\mathscr{B}$ verifies whether $(S,(M,\rho),\delta,\tilde{\beta},RK,1,1,0) \in RK^{List}$. If yes, $\mathscr{B}$ returns SK to the adversary and resets $tag_2 = 0, tag_3 = 1$. Otherwise, the simulator selects at random $\theta,\sigma,\sigma_1,\cdots,\sigma_{|S|} \in_R \mathbb{Z}_p$, $\tilde{\beta},\delta \in_R \{0,1\}^\lambda$, $\bar{K}_0,\bar{K}_1,\cdots,\bar{K}_{|S|} \in_R \mathbb{G}$. It then sets $RK_{0,1} = \bar{K}_0 \cdot h^\theta$, $RK_{0,2} = g^\theta$, $RK_1 = g^\sigma$ and for $i = 1,\cdots,|S|$, $RK_{i,2} = g^{\sigma_i}$ and $RK_{i,3} = \bar{K}_i^{\sigma_i} \cdot v^{-\sigma}$, and constructs $Int\tilde{C}$ using $\tilde{\beta}$ and δ as in the real scheme. Eventually, $\mathscr{B}$ outputs RK to $\mathscr{A}$, and adds $(S,(M,\rho),\delta,\tilde{\beta},RK,1,0,1)$ to RK^{List}.
  - Otherwise, if $(S,(M,\rho),\delta,\tilde{\beta},RK,0,1,0) \in RK^{List}$, $\mathscr{B}$ gives RK to $\mathscr{A}$, and resets $tag_2 = 0, tag_3 = 1$. Otherwise, $\mathscr{B}$ first constructs the private key SK for the attribute set S as for the private key extraction queries. The simulator then generates RK as in the real scheme, gives it to the adversary, and adds $(S,(M,\rho),\delta,\tilde{\beta},RK,0,0,1)$ to RK^{List}.

- *Online encryption oracle $O_{one}(S,(M,\rho),IntC)$:*

  The simulator checks wether Eq. 1 holds. If not, meaning that either the intermediate ciphertext $IntC$ is invalid or S does not satisfy (M,ρ), it returns $\perp$. Otherwise, $\mathscr{B}$ works as follows.

  - If S satisfies (M^*,ρ^*) and $(S,SK) \notin SK^{List}$, then the simulator first construct the re-encryption key as the second case for the re-encryption key extraction queries, re-encrypts $IntC$ and gives it to the adversary. It then adds $(S,(M,\rho),\delta,\tilde{\beta},RK,1,1,0)$ to RK^{List} and $(S,(M,\rho),CT,0,1,0)$ to ONE^{List}.
  - Otherwise, the simulator first constructs RK as the third case for the re-encryption key extraction queries, re-encrypts $IntC$ and gives it to the adversary. It then adds $(S,(M,\rho),\delta,\tilde{\beta},RK,0,1,0)$ to RK^{List} and $(S,(M,\rho),CT,1,0,0)$ to ONE^{List}.
  - Otherwise, $\mathscr{B}$ verifies whether $(\beta,\mathscr{M},s) \in H_2^{List}$ such that $C_{0,2} = h^s = g^{\hat{h}\cdot s} \cdot \prod_{\substack{j=1,\cdots,l \\ k=1,\cdots,n}} (g^{a^k/b_j^2})^{-\rho^*(j)M^*_{j,k}\cdot s}$.

    If no such tuple exists, then the simulator returns $\perp$. Otherwise, it verifies whether $(S,(M,\rho),\delta,\tilde{\beta},\perp,\perp,1,\perp) \in RK^{List}$ such that S satisfies (M^*,ρ^*). If no, the simulator selects at random $\tilde{\beta},\delta \in \{0,1\}^\lambda$, generates $Int\tilde{C}$ as in the real scheme, in order to hide δ and $\tilde{\beta}$. It then constructs $B_2 = (e(g^a,g^{a^q}) \cdot e(g,g^{\alpha'}))^{s\cdot\xi_1}$ where $\xi_1 = H_3(\delta)$. Eventually, $\mathscr{B}$ outputs $CT = ((M,\rho),C_{0,1},,B_2,\tilde{C}_0,Int\tilde{C})$, gives it to the adversary, and adds $(S,(M,\rho),\delta,\tilde{\beta},\perp,\perp,1,\perp)$ to RK^{List} and $(S,(M,\rho),CT,0,0,1)$ to ONE^{List}.

- *Ciphertext decryption oracle $O_d(S,CT)$:*

  $\mathscr{B}$ first checks if there are tuples $(\tilde{\beta},\delta,\tilde{s})$ and $(\beta,\mathscr{M},s)$ in H_2^{List} such that $\tilde{C}_0 = g^{\tilde{s}}$ and $C_{0,2} = h^s$. If not, the simulator outputs $\perp$. Otherwise, it checks whether Eq. 2 holds. It not, meaning that either $Int\tilde{C}$ is invalid or S does not satisfy (M,ρ), it outputs $\perp$. Otherwise, $\mathscr{B}$ proceeds as follows:

– If $(S,(M,\rho),\delta,\tilde{\beta},RK,1,0,1) \in RK^{List}$ or $(S,(M,\rho),CT,0,1,0) \in ONE^{List}$, it verifies

$$\begin{aligned} &\forall j \in [1,l], e(\tilde{C}_{j,1},g) \stackrel{?}{=} e(w,g)^{\tilde{\lambda}'_j} \cdot e(v,\tilde{C}_{j,3}) \\ &\forall j \in [1,l], e(\tilde{C}_{j,2},g) \stackrel{?}{=} e(u,\tilde{C}_{j,3}^{-1})^{\tilde{x}_j} \cdot e(h,\tilde{C}_{j,3}^{-1}) \end{aligned} \tag{3}$$

where given $I = \{i : \rho(i) \in S\}$ and M, it can find the vector $\vec{\tilde{w}} = \{\tilde{w}_i \in \mathbb{Z}_p\}_{i\in I}$ such that $\sum_{i\in I} \tilde{w}_i \cdot \tilde{\lambda}_i = \tilde{s}$. If the above equation does not hold, then the simulator returns $\perp$. Otherwise, it computes $C_{0,1} = g^s$ with the knowledge of s and then checks Eq. 1. If the equation does not hold, then it returns $\perp$. Otherwise, $\mathscr{B}$ recovers the random re-encryption key $RK = (S, RK_{0,1}, RK_{0,2}, RK_1, \{RK_{i,2}, RK_{i,3}\}_{i\in[1,k]}, Int\tilde{C})$ from RK^{List} and verifies the validity of B_2 as $B_2 \stackrel{?}{=} \frac{e(C_{0,1},RK_{0,1})/e(C_{0,2},RK_{0,2})}{e(w^{\sum_{i\in I} C_{i,4}\cdot w_i},RK_1)\cdot\prod_{i\in I}(e(C_{i,1},RK_1)\cdot e(C_{i,2}\cdot u^{C_{i,5}},RK_{j,2})\cdot e(C_{i,3},RK_{j,3}))^{w_i}}$ where I and w_i are defined in the **ReEnc** phase. If the previous equation does not hold, then it returns $\perp$. Otherwise, the simulator verifies whether $(R,\delta_1) \in H_1^{List}$ such that $B_1 = (\beta \| \mathscr{M}) \oplus \delta_1$ and $R = e(g,g)^{\alpha\cdot s}$. If no such tuple exists, $\mathscr{B}$ returns $\perp$. Otherwise, it outputs $\mathscr{M}$ and gives it to $\mathscr{A}$.

– Otherwise, if $(S,SK) \in SK^{List}$, the simulator recovers $\mathscr{M}$ as in the real scheme using SK. Otherwise, $\mathscr{B}$ verifies whether Eq. 2 and Eq. 3 hold. If not, it returns $\perp$. Otherwise, it verifies whether $(R,\delta_1) \in H_1^{List}$ such that $B_1 = (\beta \| \mathscr{M}) \oplus \delta_1$ and $R = e(g,g)^{\alpha\cdot s}$, and $B_2 = e(g,g)^{\alpha\cdot s\cdot \xi_1}$. If no such tuple exists and the equations do not hold, $\mathscr{B}$ returns $\perp$. Otherwise, it outputs $\mathscr{M}$ and gives it to $\mathscr{A}$.

Challenge. The challenge ciphertext is constructed as follows. The adversary gives two equal-length messages $\mathscr{M}_0$ and $\mathscr{M}_1$ to the simulator. The simulator flips a coin b and answers to $\mathscr{A}$ as follows. $IntC$ is generated as in the real scheme, meaning that $IntC = (s, B_1, C_{0,1}, C_{0,2}, \{\lambda'_j, t_j, x_j, C_{j,1}, C_{2,j}, C_{3,j}\}_{j\in[1,l]}, D)$ We suppose that the value $C_{0,2}$ is not necessarily output.

Then, $\mathscr{B}$ chooses at random $\tilde{\beta}^*, \delta^* \in_R \{0,1\}^\lambda$, issues an H_3 query on δ^* to obtain ξ_1^*. We notice that in the previous step, $(\beta, \mathscr{M}_b)$ must be issued to H_2 such that the tuple $(\beta, \mathscr{M}_b, s)$ is already stored in H_2^{List}, where $\beta \in_R \{0,1\}^\lambda$ and $s \in_R \mathbb{Z}_p$. Thus, it recovers $(\beta, \mathscr{M}_b, s)$ from H_2^{List} and computes $B_2^* = (e(g^a, g^{a^q}) \cdot e(g, g^{\alpha'}))^{s\cdot\xi_1^*}$. Moreover, $\mathscr{B}$ chooses at random $\tilde{B}_1^* \in_R \{0,1\}^{2\lambda}$ and implicitly sets $H_1(Z \cdot e(g^{\tilde{s}}, g^{\alpha'})) = \tilde{B}_1^* \oplus (\delta^* \| \tilde{\beta}^*)$ and $\tilde{C}_0^* = g^{\tilde{s}}$.

$\mathscr{B}$ can choose the secret splitting in order to cancel out the terms. For that, it chooses at random $\hat{\tilde{y}}_2, \cdots, \hat{\tilde{y}}_n$ and shares the secret using the vector $\vec{\tilde{y}} = (\tilde{s}, \tilde{s}a + \hat{\tilde{y}}_2, \cdots, \tilde{s}a^{n-1} + \hat{\tilde{y}}_n) \in Z_p^n$. We note that the secret $\tilde{s}$ and the vector $\vec{\tilde{y}}$ are properly distributed since $\tilde{s}$ is information-theoretically hidden from $\mathscr{A}$ and the $\hat{\tilde{y}}_j$'s are uniformly randomly chosen. Since $\vec{\tilde{\lambda}} = M^* \cdot \vec{\tilde{y}}$, we have that $\tilde{\lambda}_j = \sum_{i=1,\cdots,n} M_{j,i}^* \tilde{s} a^{i-1} + \sum_{i=2,\cdots,n} M_{j,i}^* \hat{\tilde{y}}_i = \sum_{i=1,\cdots,n} M_{j,i}^* \tilde{s} a^{i-1} + \hat{\tilde{\lambda}}_j$ for each row $j = 1, \cdots, l$. We note that the values $\hat{\tilde{\lambda}}_j = \sum_{i=2,\cdots,n} M_{j,i}^* \hat{\tilde{y}}_i$ are known to $\mathscr{B}$.

For each row, $\mathscr{B}$ implicitly sets $\tilde{t}_j = -\tilde{s} b_j$, which is properly distributed since the b_j's are information-

theoretically hidden from $\mathscr{A}$. Then, $\mathscr{B}$ computes the following:

$$\begin{aligned}
\tilde{C}^*_{j,1} &= w^{\tilde{\lambda}_j} v^{\tilde{t}_j} = w^{\hat{\tilde{\lambda}}_j} \prod_{i=1,\cdots,n} g^{M^*_{j,i}\tilde{s}a^i} \cdot (g^{\tilde{s}b_j})^{-\hat{v}} \cdot \prod_{\substack{i'=1,\cdots,l \\ k=1,\cdots,n}} g^{-M^*_{i',k}a^k\tilde{s}b_j/b_{i'}} \\
&= w^{\hat{\tilde{\lambda}}_j} \cdot (g^{\tilde{s}b_j})^{\hat{v}} \cdot \prod_{i=1,\cdots,n} g^{M^*_{j,i}\tilde{s}a^i} \cdot \prod_{k=1,\cdots,n} g^{M^*_{j,k}a^k\tilde{s}b_j/b_j} \cdot \prod_{\substack{i'=1,\cdots,l,i'\neq j \\ k=1,\cdots,n}} g^{-M^*_{i',k}a^k\tilde{s}b_j/b_{i'}} \\
&= w^{\hat{\tilde{\lambda}}_j} \cdot (g^{\tilde{s}b_j})^{\hat{v}} \cdot \prod_{\substack{i'=1,\cdots,l,i'\neq j \\ k=1,\cdots,n}} (g^{\tilde{s}a^k b_j/b_{i'}})^{-M^*_{i',k}} \\
\tilde{C}^*_{j,2} &= (u^{\rho^*(j)}h)^{\tilde{t}_j} = (g^{\tilde{s}b_j})^{-(\hat{u}\rho^*(j)+\hat{h})} \cdot (\prod_{\substack{i'=1,\cdots,l \\ k=1,\cdots,n}} g^{(\rho^*(j)-\rho^*(i'))M^*_{i',k}a^k/b^2_{i'}})^{-\tilde{s}b_j} \\
&= (g^{\tilde{s}b_j})^{-(\hat{u}\rho^*(j)+\hat{h})} \cdot \prod_{\substack{i'=1,\cdots,l,i'\neq j \\ k=1,\cdots,n}} (g^{\tilde{s}a^k b_j/b^2_{i'}})^{-(\rho^*(j)-\rho^*(i'))M^*_{i',k}})^{-\tilde{s}b_j} \\
C^*_{j,3} &= g^{\tilde{t}_j} = (g^{\tilde{s}b_j})^{-1}
\end{aligned}$$

Since $\tilde{t}_j = -\tilde{s}b_j$, we "raised" the exponents of the value so that they cancel out with the exponents of $w^{\tilde{\lambda}_j}$. The simulator then issues a H_5 query on $(\tilde{B}^*_1, \tilde{C}^*_0, (\tilde{C}^*_{1,1}, \tilde{C}^*_{1,2}, \tilde{C}^*_{1,3}), \cdots, (\tilde{C}^*_{l,1}, \tilde{C}^*_{l,2}, \tilde{C}^*_{l,3}), (M^*, \rho^*))$ to obtain $(\tilde{B}^*_1, \tilde{C}^*_0, (\tilde{C}^*_{1,1}, \tilde{C}^*_{1,2}, \tilde{C}^*_{1,3}), \cdots, (\tilde{C}^*_{l,1}, \tilde{C}^*_{l,2}, \tilde{C}^*_{l,3}), (M^*, \rho^*), \xi^*_3, \delta^*_3)$, and sets $\tilde{D}^* = (g^{\tilde{s}})^{\xi^*_3}$. Then, $\mathscr{B}$ sets $Int\tilde{C}^* = ((M^*, \rho^*), B^*_1, C^*_0, \{C^*_{j,1}, C^*_{j,2}, C^*_{j,3}\}_{j\in[1,l]}, D^*)$.

Finally, $\mathscr{B}$ returns the ciphertext $CT^* = ((M^*, \rho^*), C_{0,1}, B^*_2, Int\tilde{C})$ to $\mathscr{A}$.

If $Z = e(g,g)^{a^{q+1}\cdot s}$, then CT^* is a valid ciphertext. Indeed, the components corresponding to $IntC$ are valid. Since $IntC$ is encrypted to CT^* under a valid re-encryption key RK, the online encryption must be valid, meaning that the construction of B^*_2 is valid. Moreover, it is easy to see that the rest of the components are valid too. Nevertheless, if $Z = R \in \mathbb{G}_T$, then the challenge ciphertext is independent of the bit b for $\mathscr{A}$.

Phase 2. Same as in **Phase 1** but with the constraints defined in the Selective IND-CCA security at ciphertext Model.

Guess. $\mathscr{A}$ outputs a bit $b' \in_R \{0,1\}$ for b. If $b' = b$, then $\mathscr{B}$ outputs 0, meaning that it claims that the challenge Z is equal to $e(g,g)^{a^{q+1}\cdot s}$; otherwise, it outputs 1.

If $Z = e(g,g)^{a^{q+1}\cdot s}$, then $\mathscr{A}$ played the proper security game, since $B_1 = (\mathscr{M}_b \| \beta) \oplus H_1(e(g,g^s)^{\alpha'} \cdot Z) = (\mathscr{M}_b \| \beta) \oplus H_1(e(g,g)^{\alpha\cdot s})$. If Z is a random term R of $\mathbb{G}_T$, then all information about the message $\mathscr{M}_b$ is lost in the challenge ciphertext. Thus, the advantage of $\mathscr{A}$ is exactly 0. Therefore, if $\mathscr{A}$ breaks the proper security game with a non-negligible advantage, then $\mathscr{B}$ has a non-negligible advantage in breaking the $q-1$ assumption.

Analysis of the simulations of the Random Oracles

The simulations of the random oracles are perfect except H_1, H_2 and H_3. Let H^*_1 and H^*_2 be the events that $\mathscr{A}$ has queried before the **Challenge** phase: $R^* = e(g,g)^{\alpha s}$ to H_1 (the probability of successful query is $q_{H_1}/2^{2\lambda}$) and $(\beta^*, \mathscr{M}_b)$ to H_2 (the probability of successful query is q_{H_2}/p). Let H^*_3 denotes the event that $\mathscr{A}$ queried δ^* to H_3 before the **Challenge** phase: this happens with probability q_{H_3}/p. In the simulation of O_{sk}, the responses to $\mathscr{A}$ are perfect.

In the simulation of O_d, it might be possible that the simulator cannot provide a decryption for a valid ciphertext. Suppose the adversary can generate a valid ciphertext without querying $e(g,g)^{\alpha s}$ to H_1, where $s = H_2(\beta, \mathscr{M})$. Let $ValCT$ be the event that the ciphertext is valid, $QueryH_1$ be the event that the adversary has queried $e(g,g)^{\alpha s}$ to H_1, and $QueryH_2$ be the event that the adversary has queried $(\beta, \mathscr{M})$ to H_2. From the simulation, $Pr[ValCT \mid \neg QueryH_1] \leq Pr[QueryH_2 \mid QueryH_1] + Pr[ValCT \mid QueryH_1 \wedge \neg QueryH_2] \leq q_{H_2}/2^{2\lambda} + 1/p$ and $Pr[ValCT \mid \neg QueryH_2] \leq q_{H_1}/2^{2\lambda} + 1/p$, where q_{H_1}, q_{H_2} are the maximum numbers of random oracle queries to H_1, H_2. Let $Pr[ErrDec]$ be the probability that the event $ValCT \mid (\neg QueryH_1 \vee \neg QueryH_2)$ occurs, then $Pr[ErrDec] \leq (\frac{q_{H_1}+q_{H_2}}{2^{2\lambda}} + \frac{2}{p})q_d$, where q_d is the total number of decryption queries.

Let Bad denote the event that $(H_2^* \mid \neg H_1^*) \vee H_1^* \vee H_3^* \vee ErrDec$, then

$$\begin{aligned} \varepsilon_1 &= |\, Pr[b' = b] - 1/2 \,| \leq \frac{1}{2} Pr[Bad] = \frac{1}{2} Pr[(H_2^* \mid \neg H_1^*) \vee H_1^* \vee H_3^* \vee ErrDec] \\ &\leq \frac{1}{2}\left(\frac{q_{H_3} + 2q_d}{p} + \frac{q_{H_2} + (q_{H_1} + q_{H_2})q_d}{2^{2\lambda}}\right) \end{aligned}$$

Therefore, $Adv_{\mathscr{B}}^{q-1} \geq \frac{1}{q_{H_1}}(2\varepsilon_1 - \frac{q_{H_2}+(q_{H_1}+q_{H_2})q_d}{2^{2\lambda}} - \frac{q_{H_3}-2q_d}{p})$. □

5.3 Insider Attacks

Selective Collusion Resistance Proof

Theorem 2. *Suppose the CBDHE assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ and H_1, H_2, H_3, H_4 and H_5 are hash functions, our OO-CP-AB-PRE scheme is selectively collusion resistant in the random oracle model.*

Proof. Suppose there is an adversary $\mathscr{A}$ who can break the collusion resistance of our scheme. We then construct a reduction algorithm $\mathscr{B}$ to output Z equal to $e(g,g)^{abc}$. The simulator $\mathscr{B}$ plays the collusion resistance game with $\mathscr{A}$ as follows.

$\mathscr{B}$ takes as input $(p, \mathbb{G}, \mathbb{G}_T, e)$ and a CBDHE problem instance $T = (g, g^a, g^b, g^c)$; its goal is to output $Z = e(g,g)^{abc}$.

Initialization. The adversary gives the challenge set S^* to $\mathscr{B}$.

Setup. The simulator sets $w = g^b, e(g,g)^\alpha = e(g^a, g^b) = e(g,g)^{ab}$. It also chooses at random $u, h, v \in_R \mathbb{G}$ and gives these public parameters to $\mathscr{A}$.

We note that the public parameters are identical to those in the real scheme for the adversary. At any time, $\mathscr{A}$ can adaptively query the random oracles H_j for $j \in [1,5]$, which are controlled by $\mathscr{B}$. The simulator maintains the lists H_j^{List} for $j \in [1,4]$, which are initially empty, and answers the queries to the random oracles as follows.

- H_1: on receipt of an H_1 query on $R \in \mathbb{G}_T$, if there is a tuple $(R, \delta_1) \in H_1^{List}$, $\mathscr{B}$ forwards the predefined value δ_1 to $\mathscr{A}$, where $\delta_1 \in \{0,1\}^{2\lambda}$. Otherwise, $\mathscr{B}$ sets $H_1(R) = \delta_1$, responds δ_1 to $\mathscr{A}$ and adds the tuple (R, δ_1) to H_1^{List}, where $\delta_1 \in_R \{0,1\}^{2\lambda}$.
- H_2: on receipt of an H_2 query on $(\beta, \mathscr{M})$, if there is a tuple $(\beta, \mathscr{M}, s) \in H_2^{List}$, $\mathscr{B}$ forwards the predefined value s to $\mathscr{A}$, where $s \in \mathbb{Z}_p$. Otherwise, $\mathscr{B}$ sets $H_2(\beta, \mathscr{M}) = s$, responds s to $\mathscr{A}$ and adds the tuple $(\beta, \mathscr{M}, s)$ to H_2^{List}, where $s \in_R \mathbb{Z}_p$.
- H_3: on receipt of an H_3 query on $\delta \in \{0,1\}^\lambda$, if there is a tuple (δ, ξ_1) in H_3^{List}, $\mathscr{B}$ forwards the predefined value ξ_1 to $\mathscr{A}$, where $\xi_1 \in \mathbb{Z}_p$. Otherwise, $\mathscr{B}$ sets $H_3(\delta) = \xi_1$ to $\mathscr{A}$ and adds the tuple (δ, ξ_1) to H_3^{List}, where $\xi_1 \in_R \mathbb{Z}_p$.

- H_4: on receipt of an H_4 query on $(B_1, C_{0,1}, C_{0,2}, (C_{1,1}, C_{1,2}, C_{1,3}), \cdots, (C_{l,1}, C_{l,2}, C_{l,3}))$, if there is a tuple $(B_1, C_{0,1}, C_{0,2}, (C_{1,1}, C_{1,2}, C_{1,3}), \cdots, (C_{l,1}, C_{l,2}, C_{l,3}), \xi_2, \delta_2) \in H_4^{List}$, $\mathscr{B}$ forwards the predefined value δ_2 to $\mathscr{A}$, where $\xi_2 \in \mathbb{Z}_p, \delta_2 \in \mathbb{G}$. Otherwise, $\mathscr{B}$ sets $\delta_2 = g^{\xi_2}$, responds δ_2 to $\mathscr{A}$ and adds the tuple $(B_1, C_{0,1}, C_{0,2}, (C_{1,1}, C_{1,2}, C_{1,3}), \cdots, (C_{l,1}, C_{l,2}, C_{l,3}), \xi_2, \delta_2)$ in H_4^{List}, where $\xi_2 \in_R \mathbb{Z}_p$.
- H_5: on receipt of an H_5 query on $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho))$, if there is a tuple $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho), \xi_3, \delta_3) \in H_5^{List}$, $\mathscr{B}$ forwards the predefined value δ_2 to $\mathscr{A}$, where $\xi_3 \in \mathbb{Z}_p, \delta_3 \in \mathbb{G}$. Otherwise, $\mathscr{B}$ sets $\delta_3 = g^{\xi_3}$, responds δ_3 to $\mathscr{A}$ and adds the tuple $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho), \xi_3, \delta_3)$ in H_5^{List}, where $\xi_3 \in_R \mathbb{Z}_p$.

Query Phase. The simulator answers to $\mathscr{A}$'s queries as follows.

- *Private key extraction oracle $O_{sk}(S)$:*

  Let $S = \{A_1, \cdots, A_k\}$ be an attribute set such that $S \neq S^*$. Thus, there exists an index j such that $A_j \in S \wedge A_j \notin S^*$ or $A_j \notin S \wedge A_j \in S^*$. Without lost of generality, we focus on the analysis of $A_j \in S \wedge A_j \notin S^*$. For $i = 1, \cdots, k$. $\mathscr{B}$ randomly chooses $r'_i \in_R \mathbb{Z}_p$ and implicitly sets the elements r and r_i as follows:

$$\begin{aligned} r_i &= r'_i \text{ for } i \neq j \\ r_j &= -a + r'_j \\ r &= -a + \sum_{i=1}^{k} r'_i \end{aligned}$$

  Therefore, the components of the private key SK can be computed as follows:

$$\begin{aligned} &K_0 = g^{\sum_{i=1}^k r'_i} = g^{ab} \cdot g^{b(-a+\sum_{i=1}^k r'_i} && K_1 = g^{-a+\sum_{i=1}^k r'_i} \\ &K_{i,2} = g^{r'_i} && K_{i,3} = (u^{A_i} h)^{r'_i} v^{a-\sum_{i=1}^k r'_i} \text{ for } i = 1, \cdots, k,\ i \neq j \\ &K_{j,2} = g^{-a+r'_j} && K_{j,3} = (u^{A_j} h)^{-a+r'_j} v^{a-\sum_{i=1}^k r'_i} \end{aligned}$$

  $\mathscr{B}$ returns the private key $SK = (S, K_0, K_1, \{K_{i,2}, K_{i,3}\}_{i \in [1,k]})$ to $\mathscr{A}$.

- *Re-encryption key extraction oracle $O_{rk}(S)$:*

  Let S be an attribute set. If $S \neq S^*$, then $\mathscr{A}$ proceeds as follows: the private key SK is obtained from $\mathbf{KeyGen}(PK, MSK, S)$ and RK from $\mathbf{ReKeyGen}(PK, SK)$. Otherwise, it proceeds as follows.

  First, $\mathscr{B}$ computes $Int\tilde{C}$: it picks at random $\tilde{\beta}, \delta \in \{0,1\}^\lambda$ and issues a H_2 query on $(\tilde{\beta}, \delta)$ to obtain $\tilde{s}$ and a H_1 query on $e(g,g)^{\alpha \cdot \tilde{s}} = e(g,g)^{ab \cdot \tilde{s}}$ to obtain δ_1. It then computes $\tilde{B}_1 = (\delta \| \tilde{\beta}) \oplus \delta_1$ and $\tilde{C}_0 = g^{\tilde{s}}$. It also picks at random $\tilde{y}_2, \cdots, \tilde{y}_n \in \mathbb{Z}_p$, sets the vector $\vec{\tilde{y}} = (\tilde{s}, \tilde{y}_2, \cdots, \tilde{y}_n)^T$, and computes a vector of shares of $\tilde{s}$ as $(\tilde{\lambda}_1, \cdots, \tilde{\lambda}_l)^T = M \cdot \vec{\tilde{y}}$, where M is a $l \times n$ matrix and $l \leq P$. Then, for $j = 1, \cdots, l$, it chooses at random $\tilde{r}_j \in \mathbb{Z}_p$ and computes $\tilde{C}_{j,1} = w^{\tilde{\lambda}_j} v^{\tilde{r}_j}$, $\tilde{C}_{j,2} = (u^{\rho(j)} h)^{-\tilde{r}_j}$ and $\tilde{C}_{j,3} = g^{\tilde{r}_j}$.

  Moreover, it issues a H_5 query on $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho))$ to obtain $(\tilde{B}_1, \tilde{C}_0, (\tilde{C}_{1,1}, \tilde{C}_{1,2}, \tilde{C}_{1,3}), \cdots, (\tilde{C}_{l,1}, \tilde{C}_{l,2}, \tilde{C}_{l,3}), (M, \rho), \xi_3, \delta_3)$, and sets $\tilde{D} = (g^{\tilde{s}})^{\xi_3}$. Eventually, the intermediate ciphertext for the re-encryption key is $Int\tilde{C} = ((M, \rho), \tilde{B}_1, \tilde{C}_0, \{\tilde{C}_{j,1}, \tilde{C}_{2,j}, \tilde{C}_{3,j}\}_{j \in [1,l]}, \tilde{D})$.

  It then issues a H_3 query on δ to obtain ξ_1 and randomly chooses $\theta \in_R \mathbb{Z}_p$. It computes $RK_{0,1} = K_0^{\xi_1} = g^{\sum_{i=1}^k r'_i \xi_1}$, $RK_{0,2} = g^\theta$, and $RK_1 = K_1^{\xi_1} = g^{-a+\sum_{i=1}^k r'_i \xi_1}$. It also chooses an index j in $[1,k]$ and computes $RK_{i,2}$ and $RK_{i,3}$ as follows:

$$RK_{i,2} = K_{i,2}^{\xi_1} = g^{r'_i \xi_1} \qquad RK_{i,3} = K_{i,3}^{\xi_1} = ((u^{A_i} h)^{r'_i} v^{a-\Sigma_{i=1}^{k} r'_i})^{\xi_1} \text{ for } i = 1, \cdots, k,\ i \neq j$$
$$RK_{j,2} = K_{j,2}^{\xi_1} = g^{-a+r'_j \xi_1} \qquad RK_{j,3} = K_{j,3}^{\xi_1} = ((u^{A_j} h)^{-a+r'_j} v^{a-\Sigma_{i=1}^{k} r'_j})^{\xi_1}$$

$\mathscr{B}$ returns the re-encryption key $RK = (S^*, RK_{0,1}, RK_{0,2}, RK_1, \{RK_{i,2}, RK_{i,3}\}_{i\in[1,k]}, Int\tilde{C})$ to $\mathscr{A}$.

Output. The challenge private key $SK^* = (S^*, K_0^*, K_1^*, \{K_{i,2}^*, K_{i,3}^*\}_{i\in[1,k^*]})$ for $S^* = \{A_1^*, \cdots, A_{k^*}^*\}$ is constructed as follows. If SK^* is a valid key, then it should satisfy the following equation:

$$\frac{e(g, K_0^*)}{e(wv, K_1^*) \cdot \prod_{i=1}^{k^*} e(u^{A_i^*} h, K_{i,2}^*) \cdot e(g, K_{i,3}^*)} = e(g,g)^{ab} = e(g,g)^{\alpha}.$$

Then, $\mathscr{B}$ outputs:

$$\frac{e(g^c, K_0^*)}{e(wv, K_1^*) \cdot \prod_{i=1}^{k^*} e(u^{A_i^*} h, K_{i,2}^*) \cdot e(g^c, K_{i,3}^*)} = e(g,g)^{abc}$$

and solves the CBDH problem.

Analysis of the simulations of the Random Oracles

In the simulation of O_{sk}, the responses to $\mathscr{A}$ are perfect. The simulations of the random oracles are perfect except H_3. Let H_3^* be the event that $\mathscr{A}$ has queried before the **Output** phase δ to H_3: this event happens with probability q_{H_3}/p. Therefore, $Succ_{\mathscr{B}}^{CBDHE} \geq \varepsilon_2 - \frac{q_{H_3}}{p}$. □

6 Computation Cost Analysis

We discuss about the computation costs in our scheme and compare them to other schemes. We provide the results in Table 1.

The algorithm **Setup** is similar in the four proposed systems and has constant size. In all the protocols, the algorithm **KeyGen** executes $O(c)$ exponentiations in $\mathbb{G}$, where c is the maximum $max(k, l, l', |I|)$. During the run of the algorithms **OffEnc** and **Enc** in the different schemes, the number of exponentiations in $\mathbb{G}$ is linear in the number of rows of the matrix representing the access structure: in **OffEnc**, the number of exponentiations in $\mathbb{G}$ is $O(P)$, where P is the maximum bound of rows in any LSSS access structure and in **Enc**, the number of exponentiations in $\mathbb{G}$ is $O(c)$. An extra computation cost for the (offline) encryption is the cost of computing one exponentiation in $\mathbb{G}_T$.

Comparing our work and [28, 15], the number of exponentiations in $\mathbb{G}_T$ required for the algorithm **Dec** is linear in the number c; whereas in [20], the number of exponentiations in $\mathbb{G}_T$ is constant. In all the systems, the numbers of pairings performed for the decryption is linear in the number c. In [15], an extra computation cost for the decryption is about the cost of computing $O(c)$ exponentiations in $\mathbb{G}$. Comparing our work with [20], an extra computation cost for the decryption is about the cost of computing $O(1)$ exponentiations in $\mathbb{G}$.

Our scheme differs in the execution of the algorithms **ReKeyGen** and **OnEnc**. We notice that there is no computation cost for the online encryption in [15]. The algorithm **ReKeyGen** of our protocol reveals the LSSS access structure and forces to compute elements according to it; whereas in [20], an access structure is already defined in **Enc**, enabling to mitigate the computation cost in **ReKeyGen**. In the algorithm **OnEnc** of our protocol, the Eq. 1 requires the computation of $O(P)$ inversions in $\mathbb{G}$, exponentiations in $\mathbb{G}_T$ and pairings to ensure the integrity of the elements computed previously. An extra computation cost for the online encryption of our system is due to the cost of computing $O(c)$ exponentiations in $\mathbb{G}$ and in $\mathbb{G}_T$ and pairings. This cost is similar to the one in **ReEnc** in [20].

Scheme	Algorithm	Exp. in $\mathbb{G}$	Exp. in $\mathbb{G}_T$	Pairings
[28] CP-ABE	**Setup**		1	1
	KeyGen	$3+4c$		
	Enc	$1+5c$	1	
	Dec		c	$1+3c$
[15] OO-CP-ABE	**Setup**		1	1
	KeyGen	$3+4c$		
	OffEnc	$1+5P$	1	
	OnEnc			
	Dec	$1+c$	c	$2+3c$
[20] CP-AB-PRE	**Setup**	1	1	1
	KeyGen	$3+c$		
	Enc	$3+3c$	1	
	ReKeyGen	$2+3c$	1	
	ReEnc	$1+2c$	c	$9+4c$
	Dec1	1	1	$8+4c$
	Dec2	4	1	$3+2c$
Our OO-CP-AB-PRE	**Setup**		1	1
	KeyGen	$3+4c$		
	OffEnc	$3+5P$	1	
	ReKeyGen	$6+7c$	1	
	OnEnc	$1+c+2P$	$c+2P$	$9+3c+6P$
	Dec	2	$1+c$	$3+3c$

Table 1: Group operation benchmarks. Exp. designates the number of exponentiations in the groups. We include the number of inversions in $\mathbb{G}$ in the number of exponentiations in $\mathbb{G}$. A blank means that there is no group operation performed. Let **Setup** be the setup algorithm, **KeyGen** be the key generation algorithm, **Enc** be the encryption algorithm, **OffEnc** be the offline encryption algorithm, **ReKeyGen** be the re-key generation algorithm, **OnEnc** be the online encryption algorithm, **ReEnc** be the re-encryption algorithm, **Dec** be the decryption algorithm, **Dec1** be the decryption algorithm using the secret key, and **Dec2** be the decryption algorithm using the re-encryption key. Let k denote the number of attributes in the set $S = \{A_1, \cdots, A_k\}$, P denote the maximum bound of rows in any LSSS access structure used in a ciphertext, l and l' denote the number of rows in the access structures (M, ρ) and (M', ρ') respectively, and $|I|$ denote the cardinality of the set $I = \{i : \rho(i) \in S\} \subset \{1, \cdots, l\}$. For clarity in the table, let c be the maximum $max(k, l, l', |I|)$ and we suppose that P may be much more greater than c.

7 Conclusion

In this paper, we presented the first single-hop unidirectional Online/Offline Ciphertext-Policy Attribute-Based Proxy Re-Encryption (OO-CP-AB-PRE) system, proven selectively CCA secure in the random oracle model and selectively collusion resistant in the random oracle model. The scheme supports Attribute-Based Proxy Re-Encryption with any monotonic access structures and deals with Online/Offline encryption.

References

[1] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security*, 9(1):1–30, February 2006.

[2] N. Attrapadung, J. Herranz, F. Laguillaumie, B. Libert, E. de Panafieu, and C. Ràfols. Attribute-based encryption schemes with constant-size ciphertexts. *Theoretical Computer Science*, 422:15–38, March 2012.

[3] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Israel, 1996.

[4] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proc. of the 7th IEEE Symposium on Security and Privacy (SP'07), Berkeley, CA, USA*, pages 321–334. IEEE, May 2007.

[5] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In *Proc. of International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT'98), Espoo, Finland, LNCS*, volume 1403, pages 127–144. Springer-Verlag, June 1998.

[6] D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *Proc. of the International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'04), Interlaken, Switzerland, LNCS*, volume 3027, pages 223–238. Springer-Verlag, May 2004.

[7] R. W. Bradshaw, J. E. Holt, and K. E. Seamons. Concealing complex policies with hidden credentials. In *Proc. of the 11th ACM conference on Computer and communications security (CCS'04), Washington DC, USA*, pages 146–157. ACM, October 2004.

[8] R. Canetti and S. Hohenberger. Chosen-ciphertext secure proxy re-encryption. In *Proc. of the 14th ACM conference on Computer and communications security (CCS'07), Alexandria, Virginia, USA*, pages 185–194. ACM, November 2007.

[9] M. Chase. Multi-authority attribute based encryption. In *Proc. of the 4th conference on Theory of cryptography (TCC'07), Amsterdam, The Netherlands, LNCS*, pages 515–534. Springer-Verlag, February 2007.

[10] M. Chase and S. S. Chow. Improving privacy and security in multi-authority attribute-based encryption. In *Proceedings of the 16th ACM conference on Computer and communications security (CCS'09), Chicago, Illinois, USA*, pages 121–130. ACM, November 2009.

[11] L. Cheung and C. Newport. Provably secure ciphertext policy abe. In *Proceedings of the 14th ACM conference on Computer and communications security (CCS'07), Alexandria, Virginia, USA*, pages 456–465. ACM, November 2007.

[12] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, March 1996.

[13] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. of the 13th ACM conference on Computer and communications security (CCS'06), Alexandria, Virginia, USA*, pages 89–98. ACM, November 2006.

[14] F. Guo, Y. Mu, and Z. Chen. Identity-based online/offline encryption. In *Proc. of the 12th International Conference (FC'08), Cozumel, Mexico, LNCS*, volume 5143, pages 247–261. Springer-Verlag, January 2008.

[15] S. Hohenberger and B. Waters. Online/offline attribute-based encryption. In *Proc. of the 17th International Conference on Practice and Theory in Public-Key Cryptography (PKC'14), Buenos Aires, Argentina, LNCS*, volume 8383, pages 293–310. Springer-Verlag, March 2014.

[16] A. Ivan and Y. Dodis. Proxy cryptography revisited. In *Proc. of the 10th Annual Network and Distributed System Security Symposium (NDSS'03), San Diego, CA, USA*. ACM, February 2003.

[17] A. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *Proc. of the 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'12), Cambridge, UK, LNCS*, volume 7237, pages 318–335. Springer-Verlag, April 2012.

[18] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Proc. of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'10), French Riviera, France, LNCS*, pages 62–91. Springer-Verlag, June 2010.

[19] A. Lewko and B. Waters. Unbounded hibe and attribute-based encryption. In *Proc. of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'11),*

Tallinn, Estonia, LNCS, volume 6632, pages 547–567. Springer-Verlag, May 2011.

[20] K. Liang, L. Fang, W. Susilo, and D. S. Wong. A ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security. In *Proceedings of the 5th International Conference on Intelligent Networking and Collaborative Systems (INCoS'13), Xi'an, China, LNCS*, pages 552–559. Springer-Verlag, September 2013.

[21] X. Liang, Z. Cao, H. Lin, and J. Shao. Attribute based proxy re-encryption with delegating capabilities. In *Proc. of the 4th International Symposium on Information, Computer, and Communications Security (ASI-ACCS'09), Sydney, Australia*, pages 276–286. ACM, March 2009.

[22] S. Luo, J. Hu, and Z. Chen. Ciphertext policy attribute-based proxy re-encryption. In *Proceedings of the 12th international conference on Information and communications security (ICICS'10), Barcelona, Spain, LNCS*, pages 401–415. Springer-Verlag, 2010.

[23] M. Mambo and E. Okamoto. Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, E80-A(1):54–63, January 1997.

[24] G. Miklau and D. Suciu. Controlling access to published data using cryptography. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB '03), Berlin, Germany*, volume 29, pages 898–909. VLDB Endowment, 2003.

[25] T. Mizuno and H. Doi. Hybrid proxy re-encryption scheme for attribute-based encryption. In *Proc. of the 5th international conference on Information security and cryptology (Inscrypt'09), Beijing, China, LNCS*, pages 288–302. Springer-Verlag, December 2010.

[26] T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *Proc. of the 30th Annual Cryptology Conference (CRYPTO'10), Santa Barbara, CA, LNCS*, volume 6223, pages 191–208. Springer-Verlag, August 2010.

[27] R. Ostrovsky, A. Sahai, and B. Waters. Attribute-based encryption with non-monotonic access structures. In *Proc. of the 14th ACM conference on Computer and communications security (CCS'07), Alexandria, Virginia, USA*, pages 195–203. ACM, November 2007.

[28] Y. Rouselakis and B. Waters. Practical constructions and new proof methods for large universe attribute-based encryption. In *Proc. of the 20th ACM conference on Computer and communications security (CCS'13), Berlin, Germany*, pages 463–474. ACM, November 2013.

[29] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proc. of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'05), Aarhus, Denmark, LNCS*, pages 457–473. Springer-Verlag, May 2005.

[30] A. Shamir and Y. Tauman. Improved online/offline signature schemes. In *Proc. of the 21st Annual International Cryptology Conference on Advances in Cryptology (CRYPTO'01), Santa Barbara, CA, USA, LNCS*, pages 355–367. Springer-Verlag, August 2001.

[31] N. P. Smart. Access control using pairing based cryptography. In *Proc. of the 2003 RSA conference on The cryptographers' track (CT-RSA'03), San Francisco, CA, USA, LNCS*, pages 111–121. Springer-Verlag, 2003.

[32] B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Proc. of the 14th international conference on Practice and theory in public key cryptography conference on Public key cryptography (PKC'11), Taormina, Italy, LNCS*, pages 53–70. Springer-Verlag, March 2011.