Privacy Preserving Revocable Predicate Encryption Revisited

Kwangsu Lee[*] Intae Kim[†] Seong Oun Hwang[‡]

Abstract

Predicate encryption (PE) that provides both the access control of ciphertexts and the privacy of ciphertexts is a new paradigm of public-key encryption. An important application of predicate encryption is a searchable encryption system in a cloud storage, where it enables a client to securely outsource its data to an untrusted cloud server and to search over it even without revealing a keyword itself. One practical issue of predicate encryption is to devise an efficient revocation method to revoke a user when the secret key of the user is compromised. Privacy preserving revocable predicate encryption (RPE) can provide not only revocation, but also the privacy of revoked users. In this paper, we first define two new security models of privacy preserving RPE: the strongly full-hiding security and the weakly full-hiding security. The strongly full-hiding security provides the full privacy of ciphertexts against outside and inside adversaries, but the weakly full-hiding security only provides the full privacy of ciphertexts against an outside adversary who cannot decrypt the challenge ciphertext. Next, we propose two general RPE constructions from any inner product encryption (IPE) schemes, and prove their security. This first RPE scheme provides the strongly full-hiding security, but the size of ciphertexts is proportional to the number of users in the system. The second RPE scheme improves the efficiency of the first RPE scheme such that the size of ciphertexts is sublinear and the decryption algorithm is efficient, but it provides the weakly full-hiding security.

Keywords: Public-key encryption, Predicate encryption, Revocation, Privacy, Adaptive security.

1 Introduction

Predicate encryption (PE) is a special type of functional encryption and the concept of PE was proposed by Boneh and Waters [11, 16]. Public-key encryption (PKE) is suitable for traditional one-to-one communication environments since only the receiver who was specified by a sender can decrypt the ciphertext of the sender. Functional encryption, by contrast, can be used for more complex communication environments of today since any user who has a privileged secret key for specific ciphertexts can decrypt the ciphertexts [9, 14, 25, 15, 6, 11, 10]. In functional encryption, a ciphertext is associated with an attribute x and a secret key is associated with a function f, and a user who has a secret key associated with f can decrypt a ciphertext associated with x if $f(x) = 1$. Whereas functional encryption provides the message hiding property, PE provides not only the message hiding property but also the attribute hiding property.

One important application of PE is the searches on encrypted data in outsourced storages. For example, the outsourced storage of a hospital can store the ciphertext of patient's health information by

[*]Columbia University, NY, USA
[†]Hongik University, Korea
[‡]Hongik University, Korea

attaching a keyword x to the ciphertext. A researcher can request a secret key for his research area f to the authorized center and receives the secret key associated with f. After that, the researcher can access to a limited set of ciphertexts in the outsourced storage if $f(x) = 1$. In this case, the researcher cannot access to other ciphertexts such that $f(x) \neq 1$. Therefore, PE not only protects the privacy of patients, but also enables researchers to use the health information of patients.

The large obstacle of using the PE schemes in real applications is to provide an efficient revocation method that can revoke a user whose secret key was compromised. For example, if the secret key of a researcher in the outsourced storage of the hospital is leaked, then the secret key of the researcher should be revoked to prevent an adversary from using the secret key to access the information of patients. Thus revocation is a very important mechanism to preserve the whole security of the system. In PKE, an efficient revocation method was easily achieved by using the certificate revocation list (CRL) since a sender can check the validity of a receiver's certificate before he sends a ciphertext. However, it is not easy to apply the revocation method of PKE to PE since there is no user's certificate in PE. In identity-based encryption (IBE), Boneh and Franklin proposed a revocation method such that the private-key generator (PKG) issues the secret key of a user for an identity that includes an original identity and time information [9]. However, this method does not provide an efficient revocation method in PE since the attribute of PE consists of many identities and the revocation of one attribute affects many users that have the same attributes. Another revocation method is for the authorized center to publish secret key update information regularly [7]. Though this approach can solve the previous problems, it requires unrevoked users to update their secret keys regularly.

To revoke a user without updating the user's secret key, we may consider the sender-local revocation method, where the authorized center publishes a revoked list and a sender constructs a ciphertext such that only the non-revoked users can decrypt this. Nieto et al. proposed revokable predicate encryption (RPE) schemes by adapting this method [20]. The first scheme of Nieto et al. supports both the attribute hiding property and the message hiding property, but the second scheme of theirs additionally supports the privacy of revoked users. The privacy preserving RPE scheme of Nieto et al. is appealing since it hides all information in the ciphertext. However, their RPE scheme that provides the privacy of revoked users is only secure in a restricted security model, and it can only be constructed from a specific IPE scheme. Therefore, very natural, but interesting questions arise: Can we build a RPE scheme that provides the stronger privacy of revoked users? Can we construct a RPE scheme from any PE scheme? Can we improve the efficiency of the previous RPE scheme? In this paper, we address these problems and answer them in positive directions.

1.1 Our Contributions

We first introduce two new security models of the privacy preserving RPE schemes. In the security models, we divide adversaries into two types: an outside adversary who can not decrypt the challenge ciphertext and an inside adversary who can decrypt it. The first security model, called strongly full-hiding, is the best achievable security model for the privacy preserving RPE schemes. It provides adaptive security against outside and inside adversaries. The second security model, called weakly full-hiding, provides adaptive security against an outside adversary, but it does not provide the privacy of revoked users against an inside adversary.

Next we propose two privacy preserving RPE schemes that provide strongly full-hiding and weakly full-hiding respectively. The first scheme is a general REP construction from any IPE scheme, and it is secure in the strongly full-hiding model. This scheme provides stronger security, but it is not efficient since the number of ciphertext components is proportional to the number of users in the system. The second scheme is an efficient general RPE construction from any IPE scheme, but it is secure in the weakly full-hiding model. The number of ciphertext components in the second scheme is $r \log(N/r)$ and the decryption algorithm requires $r \log^2(N)$ decryption operations of the IPE scheme where r is the number of revoked users and N is the number of users in the system. Additionally, we can improve the efficiency of the decryption algorithm of the second scheme by using an anonymous hint system [5, 18]. Furthermore, we can build a RPE scheme whose security and efficiency are inherited from the underlying IPE scheme since our construction uses an IPE scheme in a black-box way. For instance, if we build our RPE scheme using the IPE scheme with constant size secret keys of Okamoto and Takashima [22], then the decryption algorithm of our RPE scheme just requires constant number of pairing operations.

1.2 Related Work

Functional Encryption. Functional encryption (FE) is a new paradigm of public-key encryption (PKE) and includes identity-based encryption (IBE), hierarchical IBE (HIBE), attribute-based encryption (ABE), and predicate encryption (PE) [9, 14, 25, 15, 11, 16, 10]. In FE, a ciphertext is associated with an attribute x and a secret key is associated with a function f. A user who has a private key for f can decrypt a ciphertext for x if $f(x) = 1$. PE is a special type of FE and it additionally provides the privacy of the attribute in ciphertexts [11]. PE includes anonymous IBE (AIBE) that supports equality queries [8, 1], hidden vector encryption (HVE) that supports conjunctive queries [11, 26, 17], and inner product encryption (IPE) that supports evaluations of polynomials [16, 3, 21, 23, 22].

Revocation in FE. A simple revocation method for IBE was proposed by Boneh and Franklin [9]. Their revocation method is for the private-key generator (PKG) of IBE to issue a user's secret key by attaching the current date to the identity of a user. If a user is revoked, then the PKG stops issuing the secret key of the user for a next date. However, this method has the problem of scalability since the PKG should be online and all users are required to update their secret keys regularly. Additionally, this method is not applicable to ABE and PE since the private key of ABE or PE is associated with multiple attributes and the revocation of one attribute affects many users who have the same attribute.

To solve the scalability problem of previous approach, Boldyreva et al. proposed another revocation method such that the PKG publicly broadcasts key update information to non-revoked users [7]. In this method, each user at first obtains an initial secret key from the PKG, and a sender constructs a ciphertext by encrypting a message with time information. After that, the PKG regularly broadcasts key update information that contains the updated secret keys of non-revoked users. If a user is not revoked, then he can decrypt the ciphertext after updating his secret key by using the key update information. If a user is revoked, then he cannot update his private key. This method can be applicable not only to IBE, but also to ABE and PE. Recently, Sahai et al. proposed an ABE scheme with revocable storage by extending this method [24]. One disadvantage of this method is that non-revoked users should update theirs secret keys periodically.

We may consider the sender-local revocation method to revoke users without affecting the secret keys of non-revoked users. In this method, the authorized center first posts a revocation list, and then a sender constructs a ciphertext by directly including revocation information. Attrapadung and Imai proposed a broadcast ABE scheme to support revocation in ABE [4]. Recently, Nieto et al. proposed revocable PE schemes to support revocation in PE [20].

Anonymous BE. Privacy preserving RPE is related with anonymous broadcast encryption since broadcast encryption can provide the revocation of users. Barth et al. proposed a private broadcast encryption scheme that has linear size ciphertexts and an efficient decryption algorithm, and proved its security under random oracle model [5]. Libert et al. proposed anonymous broadcast encryption schemes by improving the scheme of Barth et al. and proved their security without random oracles [18]. To improve the efficiency of anonymous broadcast encryption, Fazio and Perera proposed an outside-anonymous broadcast encryption scheme with sublinear size ciphertexts by weakening the security model of anonymity [13].

2 Preliminaries

In this section, we first define the predicate encryption and its security model. Next, we introduce the subset cover framework.

2.1 Predicate Encryption

Predicate encryption, introduced by Boneh and Waters [11], is a special type of public-key encryption where a ciphertext is associated with an attribute x and a secret key is associated with a predicate f. In predicate encryption, a user with a secret key for a predicate f can decrypt a ciphertext for an attribute x if $f(x) = 1$ and the attribute information in the ciphertext is not revealed to an adversary except the bit information of $f(x) = 1$.

Definition 2.1 (Predicate Encryption)**.** *A predicate encryption (PE) scheme for the class $\mathcal{F}$ of predicates over the set Σ of attributes consists of four PPT algorithms **Setup**, **GenKey**, **Encrypt**, and **Decrypt**, which are defined as follows:*

Setup(1^λ). *The setup algorithm takes as input a security parameter 1^λ and outputs a public key PK and a master secret key MK.*

GenKey(f, MK). *The key generation algorithm takes as input a predicate $f \in \mathcal{F}$ and the master secret key MK, and outputs a secret key SK_f.*

Encrypt(x, M, PK). *The encryption algorithm takes as input an attribute $x \in \Sigma$, a message $M \in \mathcal{M}$, and the public key PK, and outputs a ciphertext CT.*

Decrypt(CT, SK_f). *The decryption algorithm takes as input a ciphertext CT and a secret key SK_f, and outputs a message M or the distinguished symbol $\perp$.*

*The correctness property of PE is defined as follows: For all PK, MK generated by **Setup**, all $f \in \mathcal{F}$, any SK_f generated by **GenKey**, all $x \in \Sigma$, and any $M \in \mathcal{M}$, it is required that*

- *If $f(x) = 1$, then **Decrypt**(**Encrypt**$(x, M, PK), SK_f) = M$.*
- *If $f(x) = 0$, then **Decrypt**(**Encrypt**$(x, M, PK), SK_f) = \perp$ with all but negligible probability.*

The second condition of the correctness property in PE is not a trivial one to satisfy since the decryption algorithm of PE cannot easily check whether $f(x) = 0$ or not. That is, the decryption algorithm cannot obtain x from a ciphertext because of anonymity. One possible relaxation is to use a computational condition instead of a statistical condition. For a computational condition, we can use weak robustness of Abdalla et al. [2].

Predicate encryption provides not only the message hiding property (indistinguishability), but also the attribute hiding property (anonymity). The security model of predicate encryption was proposed by Boneh and Waters [11]. The security property of PE is defined as follows:

Definition 2.2 (Attribute-Hiding)**.** *The security notion of attribute-hiding under a chosen plaintext attack is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:*

1. ***Setup**: $\mathcal{C}$ runs **Setup** to generate a public key PK and a master secret key MK, and it gives PK to $\mathcal{A}$.*
2. ***Phase I**: $\mathcal{A}$ may adaptively request a polynomial number of secret keys for any predicates $f_1, \dots, f_{q'} \in \mathcal{F}$, and $\mathcal{C}$ gives the corresponding secret keys $SK_{f_1}, \dots, SK_{f_{q_1}}$ to $\mathcal{A}$ by running **GenKey**(f_i, MK).*
3. ***Challenge**: $\mathcal{A}$ outputs challenge attributes $x_0, x_1 \in \Sigma$ and challenge messages $M_0, M_1 \in \mathcal{M}$ subject to the following restrictions:*
   - *For all predicate f_i of secret key queries, it is required that $f_i(x_0) = f_i(x_1)$.*
   - *If there is f_i in secret key queries such that $f_i(x_0) = f_i(x_1) = 1$, then it is required that $M_0 = M_1$.*

   *$\mathcal{C}$ chooses a random bit b and gives the ciphertext CT to $\mathcal{A}$ by running **Encrypt**(x_b, M_b, PK).*
4. ***Phase II**: $\mathcal{A}$ may continue to request secret keys for additional predicates $f_{q'+1}, \dots, f_q$ subject to the same restrictions as before, and $\mathcal{C}$ gives the corresponding secrets keys to $\mathcal{A}$.*
5. ***Output**: Finally $\mathcal{A}$ outputs a bit b'.*

The advantage of $\mathcal{A}$ is defined as $Adv_{\mathcal{A}}^{PE,AH}(\lambda) = |\Pr[b = b'] - 1/2|$ where the probability is taken over all the randomness of the experiment. A PE scheme is (adaptively) attribute-hiding under a chosen plaintext attack if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the above experiment is negligible in the security parameter λ.

Inner product encryption (IPE) is a special type of predicate encryption and was introduced by Katz et al. [16]. In IPE, an attribute in a ciphertext is a vector $\vec{x}$ and a predicate in a secret key is a vector $\vec{y}$. The evaluation of a predicate and an attribute is the inner product operation of two vectors $\vec{x}$ and $\vec{y}$. That is, if $\langle \vec{x}, \vec{y} \rangle = 0$, then a user who has a secret key with $\vec{y}$ can decrypt the ciphertext with a vector $\vec{x}$. By using IPE, it is possible to construct a predicate encryption scheme that supports an equality query, conjunctive queries, disjunctive queries, polynomial evaluations, CNF, and DNF formulas [16]. Recently, Okamoto and Takashima proposed an IPE scheme with constant size secret key and proved its security in the adaptively attribute-hiding model under a standard assumption [22]. The IPE scheme of Okamoto and Takashima does not satisfy the second condition of the correctness property. We can use the technique of Boneh and Waters [11], or the transformation of Abdalla et al. [2] for weak robustness.

2.2 Subset Cover Framework

The subset cover framework was introduced by Naor et al., and it is a general method to construct an efficient revocation system [19]. In this framework, a collection $\mathcal{C}$ of subsets of universe $\mathcal{U} = \{1, \dots, N\}$ is defined where the users of the system are represented as the member of $\mathcal{U}$. Each subset $S_j \in \mathcal{C}$ is assigned a unique long-lived key, and a user is also assigned the long-lived key if the user belongs to the subset S_j. To encrypt a message, the authorized center first calculates a cover CV that consists of disjoint subset from the collection $\mathcal{C}$ that covers the set $\mathcal{U} \setminus R$ of receivers where R is the set of revoked users, and it encrypts the message using the long-lived keys in CV. If a user is not revoked, then he can decrypt the ciphertext using the long-lived keys assigned to him since there exists one subset in the cover CV where he belongs to. Naor et al. proposed two subset cover methods namely, the Complete Subtree (CS) method and the Subset Difference (SD) method. In this paper, we are interested in the CS method.

Definition 2.3 (Complete Subtree). *A complete subtree method for the universe $\mathcal{U} = \{1,\dots,N\}$ of users consists of four PPT algorithms **InitTree**, **GetPathValue**, **FindCover**, and **MatchCover**, which are defined as follows:*

InitTree(N). *The tree initialization algorithm takes as input the number N of users and outputs a full binary tree $\mathcal{T}_N$ such that a unique value is assigned to each node of the tree.*

GetPathValue$(\mathcal{T}_N, u)$. *The path value getting algorithm takes as input the full binary tree $\mathcal{T}_N$ and a user index $u \in \mathcal{U}$, and outputs a path value $PV_u = \{(T_0,V_0),\dots,(T_d,V_d)\}$ that consists of subtrees and values from the root node to the leaf node of u.*

FindCover$(\mathcal{T}_N, R)$. *The cover finding algorithm takes as input the full binary tree $\mathcal{T}_N$ and a set R of revoked users, and outputs a cover $CV_R = \{(T_1,V_1),\dots,(T_l,V_l)\}$ that covers the leaf nodes of non-revoked users $\mathcal{U} \setminus R$ where V_i is a value assigned to the root node of the subtree T_i.*

MatchCover(CV_R, PV_u). *The cover matching algorithm takes as input a cover CV_R and a path value PV_u of a user index u, and outputs a matching indexes (j,k) such that $(T_j,-) \in PV_u$ is equal to $(T_k,-) \in CV_R$ or outputs $\perp$.*

*The correctness property of CS is defined as follows: For all $\mathcal{T}_N$ generated by **InitTree**, all PV_u generated by **GetPathValue**, and any R, it is required that:*

- *If $u \notin R$, then **MatchCover**(**FindCover**$(\mathcal{T}_N, R), PV_u) = (j,k)$.*
- *If $u \in R$, then **MatchCover**(**FindCover**$(\mathcal{T}_N, R), PV_u) = \perp$.*

Naor et al. combined their CS method with a symmetric-key encryption scheme to construct a symmetric-key revocation scheme where the authorized center only can encrypt a message [19]. Dodis and Fazio extended the CS method of Naor et al. to the public-key setting, and they combined the extended CS method with an identity-based encryption (IBE) scheme to construct a public-key revocation scheme [12]. The main idea of Dodis and Fazio is to assign a unique identifier to the subset instead of a long-lived key, and to use the identifier as the identity of IBE. The extended CS method of Dodis and Fazio, namely, the public-key CS (PKCS) method, is described as follows:

PKCS.InitTree(N). This algorithm takes as input the number N of users. Let $N = 2^d$ for simplicity. It first sets a full binary tree $\mathcal{T}_N$ of depth d. It assigns a special identifier $*$ to the root node. For the child nodes of the root node, it assigns 0 to the left child node and assigns 1 to the right child node. For each internal node with an identifier ID, it assigns $ID||0$ to the left child node and assigns $ID||1$ to the right child node. It outputs the full binary tree $\mathcal{T}_N$ that has identifiers.

PKCS.GetPathValue$(\mathcal{T}_N, u)$. This algorithm takes as input the full binary tree $\mathcal{T}_N$ and a user index u. Let ID_d be the binary representation of $u-1$. There exists a leaf node x_u that has an identifier ID_d. It constructs a path value $PV_u = \{(T_0, ID_0), \dots, (T_d, ID_d)\}$ that consists of identifiers in the path from the root node to the leaf node x_u. It outputs a path value PV_u.

PKCS.FindCover$(\mathcal{T}_N, R)$. This algorithm takes as input the full binary tree $\mathcal{T}_N$ and a set R of revoked users. It outputs a cover $CV_R = \{(T_1, ID_1), \dots, (T_l, ID_l)\}$ where $\{T_i\}$ is a disjoint set of subtrees that covers $\mathcal{U} \setminus R$ and ID_i is the identifier of the root node of T_i.

PKCS.MatchCover(CV_R, PV_u). This algorithm takes input as a cover $CV_R = \{(T_1, V_1), \dots, (T_l, V_l)\}$ and a path value $PV_u = \{(T_0, V_0), \dots, (T_d, V_d)\}$. For $0 \le j \le d$ and $1 \le k \le l$, it finds indexes j and k such that $(T_j, -) \in PV_u$ is equal to $(T_k, -) \in CV_R$. If it found the indexes, then it outputs (j, k). Otherwise, it outputs $\perp$.

3 Revocable Predicate Encryption

In this section, we define privacy preserving revocable predicate encryption and introduce its new security models.

3.1 Definition

Revocable predicate encryption (RPE), introduced by Nieto et al., is a variation of PE that has an additional functionality such that a sender can specify the set of revoked users in ciphertexts [20]. In RPE, a user obtains a secret key that is associated with a predicate f and an index u from the PKG. The PKG posts a revocation list that contains the list of revoked user's indexes. After that, a sender constructs a ciphertext for an attribute x and a set R of revoked users. If a receiver has a secret key associated with a predicate f and an index u such that $(f(x) = 1) \wedge (u \notin R)$, then he can decrypt the ciphertext associated with an attribute x and a set R.

Definition 3.1 (Revocable Predicate Encryption)**.** *A revocable predicate encryption (RPE) scheme for the class $\mathcal{F}$ of predicates over a set Σ of attributes and the universe $\mathcal{U} = \{1, \dots, N\}$ of users consists of four PPT algorithms **Setup**, **GenKey**, **Encrypt**, and **Decrypt**, which are defined as follows:*

Setup$(1^\lambda, N)$. *The setup algorithm takes as input a security parameter 1^λ and the number N of users in the system, and then it outputs a public key PK and a master secret key MK.*

GenKey(f, u, MK). *The key generation algorithm takes as input a predicate $f \in \mathcal{F}$, a user index $u \in \mathcal{U}$, and the master secret key MK, and then it outputs a secret key $SK_{f,u}$.*

Encrypt(x, R, M, PK). *The encryption algorithm takes as input an attribute $x \in \Sigma$, a set $R \subseteq \mathcal{U}$ of revoked users, a message $M \in \mathcal{M}$, and the public key PK, and then it outputs a ciphertext CT.*

Decrypt$(CT, SK_{f,u})$. *The decryption algorithm takes as input a ciphertext CT and a secret key $SK_{f,u}$, and outputs a message M or the distinguished symbol $\perp$.*

*The correctness property of RPE is defined as follows: For all PK, MK generated by **Setup**, all $f \in \mathcal{F}$, all $u \in \mathcal{U}$, any $SK_{f,u}$ generated by **GenKey**, all $x \in \Sigma$, any $R \subseteq \mathcal{U}$, and any $M \in \mathcal{M}$, it is required that:*

- *If $(f(x) = 1) \wedge (u \notin R)$, then **Decrypt**(**Encrypt**$(x, R, M, PK), SK_{f,u}) = M$.*

- *If* $(f(x) = 0) \vee (u \in R)$, *then* ***Decrypt***$($***Encrypt***$(x, R, M, PK), SK_{f,u}) = \perp$ *with all but negligible probability.*

We can use weak robustness of Abdalla et al. [2] for the second condition of the correctness property.

3.2 Security Model

The security model of RPE was introduced by Nieto et al. [20]. They introduced two security models: the attribute-hiding (AH) security that does not provide the privacy of the set R of revoked users, and the full-hiding (FH) security that provides the privacy of the set R of revoked users. In this paper, we only consider the security model that provides the privacy of the set R. The full-hiding (FH) security of Nieto et al. is a restricted security model that limits the capability of an adversary. In this paper, we introduce the ideal security model of RPE, namely, the strongly full-hiding security, that does not limit the capability of the adversary.

The strongly full-hiding security provides the attribute hiding property, the revocation set hiding property, and the message hiding property against not only an outside adversary who cannot decrypt the challenge ciphertext, but also an inside adversary who can decrypt the challenge ciphertext. In this security model, an adversary may adaptively obtain a secret key associated with a predicate f and an index u. After that, the adversary outputs the challenge attributes x_0, x_1, the challenge revocation sets R_0, R_1, and the challenge messages M_0, M_1 with restrictions such that the adversary cannot request a secret key that trivially can distinguish the challenge ciphertext. That is, if an adversary requested a secret key for a predicate f and an index u that can decrypt the challenge ciphertext such that $(f(x_0) = 1) \wedge (u \notin R_0)) = (f(x_1) = 1) \wedge (u \notin R_1)) = 1$, then the adversary should output the challenge messages M_0, M_1 such that $M_0 = M_1$.

Definition 3.2 (Strongly Full-Hiding)**.** *The security notion of strongly full-hiding under a chosen plaintext attack is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:*

1. ***Setup***: *$\mathcal{C}$ runs **Setup** to generate a public key PK and a master secret key MK, and it gives PK to $\mathcal{A}$.*

2. ***Phase I***: *$\mathcal{A}$ may adaptively request a polynomial number of secret keys for any predicate $f \in \mathcal{F}$ with a user index $u \in \mathcal{U}$, and then $\mathcal{C}$ gives the corresponding secret key $SK_{f,u}$ to $\mathcal{A}$ by running* ***GenKey***(f, u, MK).

3. ***Challenge***: *$\mathcal{A}$ outputs challenge attributes $x_0, x_1 \in \Sigma$, challenge sets $R_0, R_1 \subseteq \mathcal{U}$ of revoked users of equal length, and challenge messages $M_0, M_1 \in \mathcal{M}$ subject to the following restrictions:*

    - *For all predicate f_i with a user index u_i of secret key queries, it is required that* $((f_i(x_0) = 1) \wedge (u_i \notin R_0)) = ((f_i(x_1) = 1) \wedge (u_i \notin R_1))$.
    - *If there is f_i with u_i in secret key queries such that* $((f_i(x_0) = 1) \wedge (u_i \notin R_0)) = ((f_i(x_1) = 1) \wedge (u_i \notin R_1)) = 1$, *then it is required that $M_0 = M_1$.*

    $\mathcal{C}$ chooses a random bit b and gives the ciphertext CT to $\mathcal{A}$ by running ***Encrypt***(x_b, R_b, M_b, PK).

4. ***Phase II***: *$\mathcal{A}$ may continue to request secret keys for additional predicates with user indexes subject to the same restrictions as before, and $\mathcal{C}$ gives the corresponding secrets keys to $\mathcal{A}$.*

5. ***Output***: *Finally $\mathcal{A}$ outputs a bit b'.*

The advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}_{\mathcal{A}}^{RPE,sFH}(\lambda) = |\Pr[b = b'] - 1/2|$ where the probability is taken over all the randomness of the experiment. A RPE scheme is strongly full-hiding under a chosen plaintext attack if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the above experiment is negligible in the security parameter λ.

Even though the strongly full-hiding security provides the full privacy of the set of revoked users, it is not easy to devise an efficient RPE scheme that satisfies this strong security. Thus we propose another security model, namely the weakly full-hiding security, by weakening the strongly full-hiding security. The weakly full-hiding security provides the same security level against an outside adversary. However, this security does not provide the privacy of the set of revoked users against an inside adversary. Therefore, it is required that the inside adversary outputs the challenge sets R_0, R_1 and the challenge message M_0, M_1 such that $R_0 = R_1$ and $M_0 = M_1$ in this weaker security model.

Definition 3.3 (Weakly Full-Hiding)**.** *The security notion of weakly full-hiding under a chosen plaintext attack is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$: The experiment is almost the same as the experiment of strongly full-hiding notion except that the restrictions in the challenge step are replaced as follows:*

- *For all predicate f_i with a user index u_i of secret key queries, it is required that $((f_i(x_0) = 1) \wedge (u_i \notin R_0)) = ((f_i(x_1) = 1) \wedge (u_i \notin R_1))$.*
- *If there is f_i with u_i in secret key queries such that $((f_i(x_0) = 1) \wedge (u_i \notin R_0)) = ((f_i(x_1) = 1) \wedge (u_i \notin R_1)) = 1$, then it is required that $R_0 = R_1$ and $M_0 = M_1$.*

The advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}_{\mathcal{A}}^{RPE,wFH}(\lambda) = \Pr[b = b'] - 1/2$ where the probability is taken over all the randomness of the experiment. A RPE scheme is weakly full-hiding under a chosen plaintext attack if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the above experiment is negligible in the security parameter λ.

Remark 3.4. *The full-hiding (FH) security of Nieto et al. [20] is weaker than the weakly full-hiding security of this paper since they didn't consider the security against an inside adversary.*

4 Strongly Full-Hiding RPE Construction

In this section, we propose a general RPE construction by using any IPE scheme, and prove that it is secure in the strongly full-hiding security model if the underlying IPE scheme is secure in the adaptively attribute-hiding security model.

The main idea of our general RPE construction is to build a PE scheme that supports a predicate for RPE by using the expressiveness of IPE. That is, we can build a PE scheme that support the predicate $f_{\vec{y},u}$ such that $f_{\vec{y},u}(\vec{x}, R) = (\langle \vec{x}, \vec{y} \rangle = 0) \wedge (u \notin R)$ where a secret key is associated with a vector $\vec{y}$ and an index u, and a ciphertext is associated with a vector $\vec{x}$ and a set R of revoked users. In this case, the security of IPE is easily reduced to the security of RPE.

4.1 Construction

Suppose that the vector of an IPE scheme is defined over elements in $\mathbb{Z}_p$. We first construct a PE scheme for the class of predicates $\mathcal{F} = \{f_{\vec{y},u} \mid \vec{y} = (y_1, \ldots, y_n) \in \mathbb{Z}_p^n, u \in \mathbb{Z}_p\}$ corresponding to a conjunction of an inner product query and a membership query where

$$f_{\vec{y},u}(\vec{x}, S) = \begin{cases} 1 & \text{if } (\langle \vec{x}, \vec{y} \rangle = 0 \mod p) \wedge (u \in S), \\ 0 & \text{otherwise.} \end{cases}$$

The PE scheme is described as follows:

PE.Setup$(1^\lambda, N)$: This algorithm first obtains PK' and MK' by running **IPE.Setup**(1^λ). It outputs a public key $PK = (PK', N)$ and a master secret key $MK = MK'$.

PE.GenKey$(f_{\vec{y},u}, MK)$: This algorithm takes as input a predicate $f_{\vec{y},u} = (\vec{y}, u)$ where $\vec{y} = (y_1, \ldots, y_n)$ and $u \in \mathcal{U}$, and the master secret key $MK = MK'$. It first builds a new vector $\vec{w} = (y_1, \ldots, y_n, u^N \mod p, u^{N-1} \mod p, \ldots, u^0 \mod p)$. It outputs a secret key $SK_{f_{\vec{y},u}}$ by running **IPE.GenKey**$(\vec{w}, MK')$.

PE.Encrypt(x, M, PK): This algorithm takes as input an attribute $x = (\vec{x}, S)$ where $\vec{x} = (x_1, \ldots, x_n)$ and a receiver set $S \subseteq \mathcal{U}$, a message M, and the public key $PK = (PK', N)$. It first parses $S = \{v_1, \ldots, v_l\}$. If $l < N$, then it sets a new set $S' = \{v_1, \ldots, v_l, dv_{l+1}, \ldots, dv_N\}$ where dv_i is a dummy index such that $dv_i \notin \mathcal{U}$. Next, it constructs a polynomial $h(u)$ of degree N such that $h(u) = \prod_{i=1}^{l}(u - v_i) \prod_{i=l+1}^{N}(u - dv_i) = c_N u^N + c_{N-1} u^{N-1} + \ldots, + c_1 u + c_0$. It selects a random $s \in \mathbb{Z}_p^*$ and builds a new vector $\vec{v} = (x_1, \ldots, x_n, sc_N \mod p, sc_{N-1} \mod p, \ldots, sc_0 \mod p)$ where $\{c_i\}$ are coefficients of $h(x)$ and outputs a ciphertext CT by running **IPE.Encrypt**$(\vec{v}, M, PK')$.

PE.Decrypt$(CT, SK_{f_{\vec{y},u}})$: This algorithm takes as input a ciphertext CT and a secret key $SK_{f_{\vec{y},u}}$, and outputs a message M by running **IPE.Decrypt**$(CT, SK_{f_{\vec{y},u}})$.

We now construct a RPE scheme using the above PE scheme that supports the class of predicates corresponds to conjunctions of a predicate query and a membership query. Our RPE scheme is described as follows:

RPE.Setup$(1^\lambda, N)$: This algorithm first runs **PE.Setup**(1^λ) to generate PK' and MK'. It outputs a public key as $PK = (PK', N)$ and a master secret key $MK = MK'$.

RPE.GenKey$(\vec{y}, u, MK)$: This algorithm takes as input a vector $\vec{y}$, a user index u, and the master secret key $MK = MK'$. It first sets a new predicate $f_{\vec{y},u} = (\vec{y}, u)$ such that $f_{\vec{y},u}(\vec{x}, S) = (\langle \vec{x}, \vec{y} \rangle = 0) \wedge (u \in S)$ and outputs a secret key $SK_{\vec{y},u}$ by running **PE.GenKey**$(f_{\vec{y},u}, MK')$.

RPE.Encrypt$(\vec{x}, R, M, PK)$: This algorithm takes as input a vector $vecx$, a set R of revoked users, a message M, and the public key $PK = (PK', N)$. It first sets a new attribute $x = (\vec{x}, S)$ where $S = \mathcal{U} \setminus R$, and outputs a ciphertext CT by running **PE.Encrypt**(x, M, PK').

RPE.Decrypt$(CT, SK_{\vec{y},u})$: This algorithm takes as input a ciphertext CT and a secret key $SK_{\vec{y},u}$, and outputs a message M by running **PE.Decrypt**$(CT, SK_{\vec{y},u})$.

4.2 Correctness

The correctness property of the PE scheme easily follows if we can show that $f_{\vec{y},u}(\vec{x},S) = ((\langle\vec{x},\vec{y}\rangle = 0 \mod p) \wedge (u \in S)) = 1$ if and only if $\langle\vec{v},\vec{w}\rangle = \langle\vec{x},\vec{y}\rangle + s \cdot h(u) = 0 \mod p$ where $\vec{v},\vec{w}$ are vectors and $h(u) = \prod_{i=1}^{l}(u - v_i)\prod_{i=l+1}^{N}(u - dv_i)$ of the IPE scheme. If $f_{\vec{y},u}(\vec{x},v) = 1$, then $\langle\vec{v},\vec{w}\rangle = 0 \mod p$ since $\langle\vec{x},\vec{y}\rangle = 0 \mod p$ and $\exists v_k$ such that $u = v_k$. If $\langle\vec{v},\vec{w}\rangle = 0 \mod p$, then we consider two cases such as $\langle\vec{x},\vec{y}\rangle = 0 \mod p$ or $\langle\vec{x},\vec{y}\rangle \neq 0 \mod p$. In case of $\langle\vec{v},\vec{w}\rangle = 0 \mod p$ and $\langle\vec{x},\vec{y}\rangle = 0 \mod p$, we have $f_{\vec{y},u}(\vec{x},v) = 1$ since $\exists v_k$ such that $u - v_k = 0$. In case of $\langle\vec{v},\vec{w}\rangle = 0 \mod p$ and $\langle\vec{x},\vec{y}\rangle \neq 0 \mod p$, the probability of $f_{\vec{y},u}(\vec{x},v) = 1$ is negligible since s is randomly chosen.

The correctness property of the RPE scheme easily follows since the RPE scheme is a PE scheme that supports the predicate $f_{\vec{y},u}$ such that $f_{\vec{y},u}(\vec{x},S) = (\langle\vec{x},\vec{y}\rangle = 0) \wedge (u \in S)$.

4.3 Security Analysis

Theorem 4.1. *The above RPE scheme is strongly full-hiding under a chosen plaintext attack if the IPE scheme is adaptively attribute-hiding under a chosen plaintext attack. That is, for any PPT adversary $\mathcal{A}$ for the above RPE scheme, there exists a PPT algorithm $\mathcal{B}$ for the PE scheme such that $Adv_{\mathcal{A}}^{RPE,sFH}(\lambda) \leq Adv_{\mathcal{B}}^{IPE,AH}(\lambda)$.*

Proof. We first show that the PE scheme is adaptively attribute-hiding if the IPE scheme is adaptively attribute-hiding. The security property of the PE scheme easily follows if we can show that $f_{\vec{y},u}(\vec{x},S) = ((\langle\vec{x},\vec{y}\rangle = 0 \mod p) \wedge (u \in S)) = 1$ if and only if $\langle\vec{v},\vec{w}\rangle = \langle\vec{x},\vec{y}\rangle + s \cdot h(u) = 0 \mod p$ where $\vec{v},\vec{w}$ are vectors and $h(u) = \prod_{i=1}^{l}(u - v_i)\prod_{i=l+1}^{N}(u - dv_i)$ of the IPE scheme. This was shown in the correctness property.

Next we show that the RPE scheme is strongly full-hiding if the PE scheme is adaptively attribute-hiding. The security of the RPE scheme easily follows since the RPE scheme is a PE scheme that supports the predicate $f_{\vec{y},u}$ such that $f_{\vec{y},u}(\vec{x},S) = (\langle\vec{x},\vec{y}\rangle = 0) \wedge (u \in S)$. This completes our proof. □

4.4 Discussions

Efficiency. Let N be the number of users and n be the size of vectors for inner product operations in the RPE scheme. In this case, the underlying IPE scheme of the RPE scheme should support vectors of size $n + N$ for inner product operations. The size of secret keys, the size of ciphertexts, and the decryption operations of the RPE scheme is the same as the size of secret keys, the size of ciphertexts, and the decryption operations of the IPE scheme respectively. For instance, if we use the IPE scheme with constant-size secret keys of Okamoto and Takashima [22], then the public key consists of $O(n+N)$ group elements, the secret key consists of 11 group elements, the ciphertext consists of $5(n+N)+1$ group elements, and the decryption algorithm requires $(n+N)$ exponentiations and 11 pairing operations.

5 Weakly Full-Hiding RPE Construction

In this section, we propose an efficient general RPE construction by using any IPE scheme, and prove that it is secure in the weakly full-hiding security model if the underlying IPE scheme is secure in the adaptively attribute-hiding security model.

The main idea of our efficient general RPE construction is to combine an IPE scheme with the public-key CS (PKCS) method. To provide the privacy of revoked users, we hide the identifier of PKCS by using the attribute hiding property of IPE. Additionally, we use a dummy identifier to hide the size of covering subsets of PKCS since two covering subsets of PKCS have different sizes even if the number of revoked users is the same. Though this construction provide the weakly full-hiding security, it does not provide the strongly full-hiding security since an inside adversary can easily obtain partial information of the revoked set through the decryption process.

5.1 Construction

Suppose that the vector of an IPE scheme is defined over elements in $\mathbb{Z}_p$. We first construct a PE scheme for the class of predicates $\mathcal{F} = \{f_{\vec{y},u} \mid \vec{y} = (y_1, \dots, y_n) \in \mathbb{Z}_p^n, u \in \mathbb{Z}_p\}$ corresponding to conjunctions of an inner product query and an equality query where

$$f_{\vec{y},u}(\vec{x}, v) = \begin{cases} 1 & \text{if } (\langle \vec{x}, \vec{y} \rangle = 0 \mod p) \wedge (u = v), \\ 0 & \text{otherwise.} \end{cases}$$

The PE scheme is described as follows:

PE.Setup(1^λ): This algorithm first obtains PK' and MK' by running **IPE.Setup**(1^λ). It outputs a public key $PK = PK'$ and a master secret key $MK = MK'$.

PE.GenKey$(f_{\vec{y},u}, MK)$: This algorithm takes as input a predicate $f_{\vec{y},u} = (\vec{y}, u)$ where $\vec{y} = (y_1, \dots, y_n)$ and a value u, and the master secret key $MK = MK'$. It first builds a new vector $\vec{w} = (y_1, \dots, y_n, u, 1)$. It outputs a secret key $SK_{f_{\vec{y},u}}$ by running **IPE.GenKey**$(\vec{w}, MK')$.

PE.Encrypt(x, M, PK): This algorithm takes as input an attribute $x = (\vec{x}, v)$ where $\vec{x} = (x_1, \dots, x_n)$ and v is a value, a message M, and the public key $PK = PK'$. It chooses a random $s \in \mathbb{Z}_p^*$ and builds a new vector $\vec{v} = (x_1, \dots, x_n, s, -sv \mod p)$ and outputs a ciphertext CT by running **IPE.Encrypt**$(\vec{v}, M, PK')$.

PE.Decrypt$(CT, SK_{f_{\vec{y},u}})$: This algorithm takes as input a ciphertext CT and a secret key $SK_{f_{\vec{y},u}}$, and outputs a message M by running **IPE.Decrypt**$(CT, SK_{f_{\vec{y},u}})$.

We now construct a RPE scheme using the above PE scheme that supports the class of predicates corresponds to conjunctions of an inner product query and an equality query. Let $N = 2^d$ be the number of users in the system, R be the set of revoked users where $|R| = r$, and L be a constant such that $L = \lceil r \log(N/r) \rceil$. We set $\mathcal{T}_N$ as the full binary tree that is defined by **PKCS.InitTree**(N). Our RPE scheme is described as follows:

RPE.Setup$(1^\lambda, N)$: This algorithm first runs **PE.Setup**(1^λ) to generate PK' and MK'. It outputs a public key as $PK = (PK', N)$ and a master secret key $MK = MK'$.

RPE.GenKey$(\vec{y}, u, MK)$: This algorithm takes as input a vector $\vec{y}$, a user index u, and the master secret key $MK = MK'$. It proceeds as follows:

1. It obtains a path value $PV_u = \{(T_0, ID_0), \dots, (T_d, ID_d)\}$ by running **PKCS.GetPathValue**$(\mathcal{T}_N, u)$.
2. For $0 \le j \le d$, it sets a new predicate $f_{\vec{y},ID_j} = (\vec{y}, ID_j)$ and computes a subkey K_j by running **PE.GenKey**$(f_{\vec{y},ID_j}, MK')$ where $f_{\vec{y},ID_j}(\vec{x}, ID) = (\langle \vec{x}, \vec{y} \rangle = 0) \wedge (ID_j = ID)$.
3. It outputs a secret key as $SK_{\vec{y},u} = \big(K_0, \dots, K_d\big)$.

RPE.Encrypt$(\vec{x}, R, M, PK)$: This algorithm takes as input a vector $\vec{x}$, a set R of revoked users, a message M, and the public key $PK = (PK', N)$. Let *dummy* be a dummy identifier and $\tilde{m}$ be a random message. It proceeds as follows:

1. It obtains a cover $CV_R = \{(T_1, ID_1), \ldots, (T_l, ID_l)\}$ by running **PKCS.FindCover**$(\mathcal{T}_N, R)$. Note that $l \leq r \log(N/r) \leq L$ by the claim of Naor et al. in [19].
2. For $1 \leq j \leq l$, it computes C_j by running **PE.Encrypt**$((\vec{x}, ID_j), M, PK')$.
3. For $l+1 \leq j \leq L$, it computes C_j by running **PE.Encrypt**$((\vec{x}, dummy), \tilde{m}, PK')$.
4. It outputs a ciphertext as $CT = \left(C_{\pi(1)}, \ldots, C_{\pi(L)}\right)$ where π is a random permutation over $\{1, \ldots, L\}$.

RPE.Decrypt$(CT, SK_{\vec{y},u})$: This algorithm takes as input a ciphertext $CT = (C_1, \ldots, C_L)$ and a secret key $SK_{\vec{y},u} = (K_0, \ldots, K_d)$. It proceeds as follows:

1. For $0 \leq j \leq d$ and $1 \leq k \leq L$, it computes M by running **PE.Decrypt**(C_k, K_j) and returns M if $M \neq \perp$.
2. If it fails to obtain $M \neq \perp$ during the above iteration, then it outputs $\perp$.

5.2 Correctness

The correctness property of the PE scheme easily follows if we can show that $f_{\vec{y},u}(\vec{x}, v) = ((\langle \vec{x}, \vec{y} \rangle = 0 \mod p) \wedge (u = v)) = 1$ if and only if $\langle \vec{v}, \vec{w} \rangle = \langle \vec{x}, \vec{y} \rangle + s \cdot (u - v) = 0 \mod p$ where $\vec{v}, \vec{w}$ are vectors of the IPE scheme. If $f_{\vec{y},u}(\vec{x}, v) = 1$, then $\langle \vec{v}, \vec{w} \rangle = 0 \mod p$ since $\langle \vec{x}, \vec{y} \rangle = 0 \mod p$ and $u = v$. If $\langle \vec{v}, \vec{w} \rangle = 0 \mod p$, then we consider two cases such as $\langle \vec{x}, \vec{y} \rangle = 0 \mod p$ or $\langle \vec{x}, \vec{y} \rangle \neq 0 \mod p$. In case of $\langle \vec{v}, \vec{w} \rangle = 0 \mod p$ and $\langle \vec{x}, \vec{y} \rangle = 0 \mod p$, we have $f_{\vec{y},u}(\vec{x}, v) = 1$ since $u - v = 0$. In case of $\langle \vec{v}, \vec{w} \rangle = 0 \mod p$ and $\langle \vec{x}, \vec{y} \rangle \neq 0 \mod p$, the probability of $f_{\vec{y},u}(\vec{x}, v) = 1$ is negligible since s is randomly chosen.

For the correctness property of the RPE scheme, we consider two conditions. If $(\langle \vec{x}, \vec{y} \rangle = 0) \wedge (u \notin R)$, then we have $(\langle \vec{x}, \vec{y} \rangle = 0) \wedge (ID_j = ID_k)$ since there exits a tuple (j, k) by the correctness property of PKCS. Thus we obtain **Decrypt**$($**Encrypt**$(\vec{x}, R, M, PK), SK_{\vec{y},u}) = M$ since **PE.Decrypt**$(C_k, K_j) = M$. If $(\langle \vec{x}, \vec{y} \rangle \neq 0) \vee (u \in R)$, then we have $(\langle \vec{x}, \vec{y} \rangle \neq 0) \vee (ID_j \neq ID_k)$ for all j and k since there is no tuple (j, k) by the correctness property of PKCS. Thus we obtain **Decrypt**$($**Encrypt**$(\vec{x}, R, M, PK), SK_{\vec{y},u}) = \perp$ since **PE.Decrypt**$(C_k, K_j) = \perp$ for all j and k by the correctness property of the above PE scheme.

5.3 Security Analysis

Theorem 5.1. *The above RPE scheme is weakly full-hiding under a chosen plaintext attack if the IPE scheme is adaptively attribute-hiding under a chosen plaintext attack. That is, for any PPT adversary $\mathcal{A}$ for the above RPE scheme, there exists a PPT algorithm $\mathcal{B}$ for the IPE scheme such that $Adv_{\mathcal{A}}^{RPE,wFH}(\lambda) \leq 2L \cdot Adv_{\mathcal{B}}^{IPE,AH}(\lambda)$ where $L = \lceil r \log(N/r) \rceil$.*

Proof. We first show that the PE scheme is adaptively attribute-hiding if the IPE scheme is adaptively attribute-hiding. The security of the PE scheme easily follows if we can show that $f_{\vec{y},u}(\vec{x}, v) = ((\langle \vec{x}, \vec{y} \rangle = 0 \mod p) \wedge (u = v)) = 1$ if and only if $\langle \vec{v}, \vec{w} \rangle = \langle \vec{x}, \vec{y} \rangle + s \cdot (u - v) = 0 \mod p$ where $\vec{v}, \vec{w}$ are vectors of the IPE scheme. This was shown in the correctness property. Thus we have $\mathsf{Adv}^{PE,AH}(\lambda) \leq \mathsf{Adv}^{IPE,AH}(\lambda)$.

Next we show that the RPE scheme is weakly full-hiding if the PE scheme is adaptively attribute-hiding. To prove the security of the RPE scheme, we divide the behavior of an adversary as two types:

type-I and type-II. The type-I adversary behaves like an outside adversary who cannot request a secret key that can decrypt the challenge ciphertext. The type-II adversary behaves like an inside adversary who can request a secret key that can decrypt the challenge ciphertext. The two types of adversaries are formally defined as follows:

Type-I. The type-I adversary $\mathcal{A}_I$ only requests a secret key for a a vector $\vec{y}_i$ and an index u_i such that $((\langle \vec{x}_0, \vec{y}_i \rangle = 0) \wedge (u_i \notin R_0)) = ((\langle \vec{x}_1, \vec{y}_i \rangle = 0) \wedge (u_i \notin R_1)) = 0$.

Type-II. The type-II adversary $\mathcal{A}_{II}$ requests at least one secret key for a a vector $\vec{y}_i$ and an index u_i such that $((\langle \vec{x}_0, \vec{y}_i \rangle = 0) \wedge (u_i \notin R_0)) = ((\langle \vec{x}_1, \vec{y}_i \rangle = 0) \wedge (u_i \notin R_1)) = 1$. Thus, the output of $\mathcal{A}_{II}$ in the challenge step should satisfy the requirements such that $R_0 = R_1$ and $M_0 = M_1$.

Let T_I, T_{II} be the event such that an adversary behave like the type-I, type-II adversary respectively. In Lemma 5.2, we show that a type-I adversary can be used for attacking the PE scheme. In Lemma 5.4, we show that a type-II adversary can be used for attacking the PE scheme too. Therefore we have

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{A}}^{RPE,wFH}(\lambda) &\leq \Pr[T_I] \cdot \mathsf{Adv}_{\mathcal{A}_I}^{RPE,wFH}(\lambda) + \Pr[T_{II}] \cdot \mathsf{Adv}_{\mathcal{A}_{II}}^{RPE,wFH}(\lambda) \\
&\leq \Pr[T_I] \cdot 2L \cdot \mathsf{Adv}_{\mathcal{B}}^{PE,AH}(\lambda) + (1 - \Pr[T_I]) \cdot L \cdot \mathsf{Adv}_{\mathcal{B}}^{PE,AH}(\lambda) \\
&\leq 2L \cdot \mathsf{Adv}_{\mathcal{B}}^{IPE,AH}(\lambda).
\end{aligned}$$

This completes our proof. □

5.3.1 Type-I Adversary

Lemma 5.2. *If there is a type-I adversary $\mathcal{A}_I$ for the above RPE scheme, then there is a PPT algorithm B for the PE scheme such that $Adv_{\mathcal{A}_I}^{RPE,wFH}(\lambda) \leq 2L \cdot Adv_{\mathcal{B}}^{PE,AH}(\lambda)$ where $L = \lceil r \log(N/r) \rceil$.*

Proof. The main idea of this proof is that a type-I adversary (that is, an outside adversary) cannot distinguish the changes of the challenge ciphertext from the correct encryption to the encryption of a random vector, a dummy set, and a random message since the type-I adversary cannot request a secret key query that can decrypt the challenge ciphertext. Thus, we first changes the challenge ciphertext from the encryption of a vector $\vec{x}_0$, a set R_0, and a message M_0 to the encryption of a random vector $\tilde{x}$, a dummy set, and a random message $\tilde{m}$ through hybrid games. Next, we also similarly changes the challenge ciphertext from the encryption of a random vector $\tilde{x}$, a dummy set, and a random message $\tilde{m}$ to the encryption of a vector $\vec{x}_1$, a set R_1, and a message M_1 through hybrid games. Therefore, if the adversary cannot distinguish these changes of hybrid games, then he cannot distinguish the changes from the encryption of $\vec{x}_0, R_0$, and M_0 to the encryption of $\vec{x}_1, R_1$, and M_1.

We first define a sequence of games $\mathbf{G}_{I,0}^0, \ldots, \mathbf{G}_{I,L}^0, \mathbf{G}_{I,L}^1, \ldots, \mathbf{G}_{I,0}^1$. The games $\mathbf{G}_{I,0}^0$ and $\mathbf{G}_{I,0}^1$ will be the original security game except that the challenge bit is fixed to 0 and 1 respectively, and the games $\mathbf{G}_{I,L}^0$ and $\mathbf{G}_{I,L}^1$ will be an ideal game such that an adversary has no advantage. Let $\tilde{x}$ be a random attribute, *dummy* be a dummy identifier, and $\tilde{m}$ be a random message. For $\gamma \in \{0,1\}$, the games are defined as follows:

Game $\mathbf{G}_{I,0}^{\gamma}$. This game is the original security game in Definition 3.3 except that the challenge bit b is fixed to γ.

Game $\mathbf{G}_{I,h}^{\gamma}$. We define a new game $\mathbf{G}_{I,h}^{\gamma}$ for $1 \leq h \leq L-1$. This game is almost identical to $\mathbf{G}_{I,h-1}^{\gamma}$ except that the ciphertext component C_h is the encryption of $(\tilde{x}, dummy)$ and $\tilde{m}$, instead of the encryption of $(\vec{x}_b, ID_h)$ and M_b. The challenge bit b is set as γ and the challenge ciphertext is constructed as follows:

1. It obtains a cover $CV_b = \{(T_1, ID_1), \ldots, (T_{l_b}, ID_{l_b})\}$ by running **PKCS.FindCover**$(\mathcal{T}_N, R_b)$. Note that $l_0, l_1 \le L$ since $|R_0| = |R_1| = r$.
2. If $1 \le h \le l_b$, then it proceeds the follows steps.
   (a) For $1 \le j \le h$, it computes C_j by running **PE.Encrypt**$((\tilde{x}, dummy), \tilde{m}, PK')$.
   (b) For $h+1 \le j \le l_b$, it computes C_j by running **PE.Encrypt**$((\vec{x}_b, ID_j), M_b, PK')$.
   (c) For $l_b+1 \le j \le L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_b, dummy), \tilde{m}, PK')$.
3. Otherwise (that is, if $l_b+1 \le h \le L$), it proceeds the follows steps.
   (a) For $1 \le j \le h$, it computes C_j by running **PE.Encrypt**$((\tilde{x}, dummy), \tilde{m}, PK')$.
   (b) For $h+1 \le j \le L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_b, dummy), \tilde{m}, PK')$.
4. It outputs a ciphertext as $CT = \big(C_{\pi(1)}, \ldots, C_{\pi(L)}\big)$ where π is a random permutation.

Game $\mathbf{G}^{\gamma}_{I,L}$. In this game $\mathbf{G}^{\gamma}_{I,L}$, the challenge ciphertext is constructed as follows:

1. For $1 \le j \le L$, it computes C_j by running **PE.Encrypt**$((\tilde{x}, dummy), \tilde{m}, PK')$.
2. It outputs a ciphertext as $CT = \big(C_{\pi(1)}, \ldots, C_{\pi(L)}\big)$ where π is a random permutation.

Let $\mathsf{Adv}^{G^{\gamma}_{I,h}}_{\mathcal{A}_I}$ be the advantage of $\mathcal{A}_I$ in $\mathbf{G}^{\gamma}_{I,h}$. That is, $\mathsf{Adv}^{G^{\gamma}_{I,h}}_{\mathcal{A}_I} = \big|\Pr[b' = b | b = \gamma] - 1/2\big|$ where b is the challenge bit and b' is the output of an adversary. We easily obtain $\mathsf{Adv}^{G^{0}_{I,h}}_{\mathcal{A}_I} = \mathsf{Adv}^{G^{1}_{I,h}}_{\mathcal{A}_I}$ for all $0 \le h \le L$ since the challenge bit is randomly chosen. We also obtain $\mathsf{Adv}^{G^{0}_{I,L}}_{\mathcal{A}_I} = \mathsf{Adv}^{G^{1}_{I,L}}_{\mathcal{A}_I} = 0$ since the challenge ciphertext is not correlated with b. In Lemma 5.3, we prove that it is hard to distinguish $\mathbf{G}^{\gamma}_{I,h-1}$ from $\mathbf{G}^{\gamma}_{I,h}$ if the underlying PE scheme is secure. Thus, we have that

$$\begin{aligned}\mathsf{Adv}^{G^{0}_{I,0}}_{\mathcal{A}_I} &= \mathsf{Adv}^{G^{0}_{I,0}}_{\mathcal{A}_I} + \sum_{h=1}^{L-1}\big(\mathsf{Adv}^{G^{0}_{I,h}}_{\mathcal{A}_I} - \mathsf{Adv}^{G^{0}_{I,h}}_{\mathcal{A}_I}\big) - \mathsf{Adv}^{G^{0}_{I,L}}_{\mathcal{A}_I} \le \sum_{h=1}^{L}\big|\mathsf{Adv}^{G^{0}_{I,h-1}}_{\mathcal{A}_I} - \mathsf{Adv}^{G^{0}_{I,h}}_{\mathcal{A}_I}\big| \\ &\le 2L \cdot \mathsf{Adv}^{PE,AH}_{\mathcal{B}}(\lambda).\end{aligned}$$

Finally, we can obtain the inequality relation as

$$\begin{aligned}\mathsf{Adv}^{RPE,wFH}_{\mathcal{A}_I}(\lambda) &= \Pr[b=0] \cdot \big|\Pr[b' = b | b = 0] - 1/2\big| + \Pr[b=1] \cdot \big|\Pr[b' = b | b = 1] - 1/2\big| \\ &= 1/2 \cdot \mathsf{Adv}^{G^{0}_{I,0}}_{\mathcal{A}_I} + 1/2 \cdot \mathsf{Adv}^{G^{1}_{I,0}}_{\mathcal{A}_I} = \mathsf{Adv}^{G^{0}_{I,0}}_{\mathcal{A}_I} \le 2L \cdot \mathsf{Adv}^{PE,AH}_{\mathcal{B}}(\lambda).\end{aligned}$$

This completes our proof. □

Lemma 5.3. *If there is a type-I adversary that distinguishes between $\mathbf{G}^{\gamma}_{I,h-1}$ and $\mathbf{G}^{\gamma}_{I,h}$ with non-negligible advantage, then there is a PPT algorithm $\mathcal{B}$ for the PE scheme such that $\big|Adv^{G^{\gamma}_{I,h-1}}_{\mathcal{A}_I} - Adv^{G^{\gamma}_{I,h}}_{\mathcal{A}_I}\big| \le 2 \cdot Adv^{PE,AH}_{\mathcal{B}}(\lambda)$.*

Proof. The basic idea of this proof is that a simulator can use the challenge oracle of PE to construct the challenge ciphertext component C_h of RPE since the only difference between $G^{\gamma}_{I,h-1}$ and $G^{\gamma}_{I,h}$ is the challenge ciphertext component C_h. Suppose there exists a type-I adversary $\mathcal{A}_I$ that distinguishes between $\mathbf{G}^{\gamma}_{I,h-1}$ and $\mathbf{G}^{\gamma}_{I,h}$ with non-negligible advantage ε. A simulator $\mathcal{B}$ that attacks the PE scheme is first given: a challenge public key PK'. Then $\mathcal{B}$ that interacts with $\mathcal{A}_I$ is described as follows:

Setup: $\mathcal{B}$ first sets $PK = (PK', N)$ and gives PK to $\mathcal{A}_I$.

Phase I: $\mathcal{A}_I$ adaptively requests a secret key for a vector $\vec{y}$ and a user index u. $\mathcal{B}$ proceeds the secret key query as follows:

1. It obtains a path value $PV_u = \{(T_0, ID_0), \dots, (T_d, ID_d)\}$ by running **PKCS.GetPathValue**$(\mathcal{T}_N, u)$.
2. For $0 \leq j \leq d$, it sets a new predicate $f_{\vec{y}, ID_j} = (\vec{y}, u)$ and requests a secret key K_j for the new predicate $f_{\vec{y}, ID_j}$ to the key generation oracle that simulates **PE.GenKey**.
3. It sets the secret key $SK_{f,u} = (K_0, \dots, K_d)$ and gives this to $\mathcal{A}_I$.

Challenge: $\mathcal{A}_I$ outputs challenge vectors $\vec{x}_0, \vec{x}_1 \in \mathbb{Z}_p^n$, challenge sets $R_0, R_1 \subseteq \mathcal{U}$, and challenge messages $M_0, M_1 \in \mathcal{M}$ subject to the restrictions of the type-I adversary. Let $\tilde{x}$ be a random vector, *dummy* be a dummy identifier, and $\tilde{m}$ be a random message. $\mathcal{B}$ sets $b = \gamma$ and proceeds as follows:

1. It obtains a cover $CV_b = \{(T_1, ID_1), \dots, (T_{l_b}, ID_{l_b})\}$ by running **PKCS.FindCover**$(\mathcal{T}_N, R_b)$.
2. If $1 \leq h \leq l_b$, then it proceeds the following steps.
   (a) For $1 \leq j \leq h-1$, it computes C_j by running **PE.Encrypt**$((\tilde{x}, dummy), \tilde{m}, PK')$.
   (b) For $j = h$, it gives challenge attributes $x'_0 = (\vec{x}_b, ID_h), x'_1 = (\tilde{x}, dummy)$ and challenge messages $M'_0 = M_b, M'_1 = \tilde{m}$ to the challenge oracle that simulates **PE.Encrypt**, and receives a challenge ciphertext C'. It sets $C_h = C'$.
   (c) For $h+1 \leq j \leq l_b$, it computes C_j by running **PE.Encrypt**$((\vec{x}_b, ID_j), M_b, PK')$.
   (d) For $l_b + 1 \leq j \leq L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_b, dummy), \tilde{m}, PK')$.
3. Otherwise (that is, if $l_b + 1 \leq h \leq L$), it proceeds the following steps.
   (a) For $1 \leq j \leq h-1$, it computes C_j by running **PE.Encrypt**$((\tilde{x}, dummy), \tilde{m}, PK')$.
   (b) For $j = h$, it gives challenge attributes $x'_0 = (\vec{x}_b, dummy), x'_1 = (\tilde{x}, dummy)$ and challenge messages $M'_0 = \tilde{m}, M'_1 = \tilde{m}$, and receives a challenge ciphertext C'. It sets $C_h = C'$.
   (c) For $h+1 \leq j \leq L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_b, dummy), \tilde{m}, PK')$.
4. It outputs a ciphertext as $CT = (C_{\pi(1)}, \dots, C_{\pi(L)})$ where π is a random permutation.

Phase II: $\mathcal{A}_I$ may continue to request secret keys for vectors and user indexes subject to the same restrictions as before, and $\mathcal{B}$ gives the corresponding secrets keys to $\mathcal{A}_I$.

Output: Finally $\mathcal{A}_I$ outputs a bit b'. $\mathcal{B}$ sets $c' = b'$ and outputs c'.

To finish the proof, we show that the distribution of the simulation is correct. It is easy to check that the public key and the secret keys are correctly distributed. Let c be the challenger oracle's random bit of the PE scheme. If $c = 0$ and $h \leq l_b$, then C' is the encryption of an attribute $x'_0 = (\vec{x}_b, ID_b)$ and a message M_b. If $c = 0$ and $h \geq l_b + 1$, then C' is the encryption of an attribute $x'_0 = (\vec{x}_b, dummy)$ and a message $\tilde{m}$. Thus, the challenge ciphertext is the same as $\mathbf{G}^{\gamma}_{I,h-1}$ if $c = 0$. Similarly, we obtain that the challenge ciphertext is the same as $\mathbf{G}^{\gamma}_{I,h}$ if $c = 1$. Therefore, we have

$$\begin{aligned}
\mathsf{Adv}^{PE,AH}_{\mathcal{B}}(\lambda) &= \Pr[c=0] \cdot \big|\Pr[c'=c|c=0] - 1/2\big| + \Pr[c=1] \cdot \big|\Pr[c'=c|c=1] - 1/2\big| \\
&= 1/2 \cdot \big|\Pr[b'=0|c=0] - 1/2\big| + 1/2 \cdot \big|\Pr[b'=1|c=1] - 1/2\big| \\
&= 1/2 \cdot \big|\Pr[b'=0|c=0] - 1/2\big| + 1/2 \cdot \big|(1-\Pr[b'=0|c=1]) - 1/2\big| \\
&\geq 1/2 \cdot (\Pr[b'=0|c=0] - 1/2) - 1/2 \cdot (\Pr[b'=0|c=1] - 1/2) \\
&= 1/2 \cdot (\Pr[b'=b|b=\gamma, c=0] - 1/2) - 1/2 \cdot (\Pr[b'=b|b=\gamma, c=1] - 1/2) \\
&= 1/2 \cdot \mathsf{Adv}^{G^{\gamma}_{I,h-1}}_{A_I} - 1/2 \cdot \mathsf{Adv}^{G^{\gamma}_{I,h}}_{A_I}.
\end{aligned}$$

This completes our proof. $\square$

5.3.2 Type-II Adversary

Lemma 5.4. *If there is a type-II adversary $\mathcal{A}_{II}$ for the above RPE scheme, then there exists a PPT algorithm B for the PE scheme such that $Adv^{RPE,wFH}_{\mathcal{A}_{II}}(\lambda) \leq L \cdot Adv^{PE,AH}_{\mathcal{B}}(\lambda)$.*

Proof. The main idea of this proof is that a type-II (that is, an inside adversary) cannot distinguish the changes of the challenge ciphertext from the encryption of $\vec{x}_0, R$, and M to the encryption of $\vec{x}_1, R$, and M through hybrid games since the type-II adversary has the restriction of $R_0 = R_1 = R$ and $M_0 = M_1 = M$. Note that we cannot change the challenge ciphertext to the encryption of a random vector, a dummy set, and a random message like the proof of the type-I adversary since the type-II adversary can easily distinguish this changes by decrypting the challenge ciphertext.

We first define a sequence of games $\mathbf{G}_{II,0}, \ldots, \mathbf{G}_{II,L}$. Let $R_0 = R_1 = R$ and $M_0 = M_1 = M$ since it is a type-II adversary. We formally define the games as follows:

Game $\mathbf{G}_{II,0}$. This game is the original security game except that the challenger bit b is fixed to 0.

Game $\mathbf{G}_{II,h}$. We define a new game $\mathbf{G}_{II,h}$ for $1 \leq h \leq L-1$. This game is almost identical to $\mathbf{G}_{II,h-1}$ except that the challenge ciphertext component C_h is changed from the encryption of $\vec{x}_0, R$, and M to the encryption of $\vec{x}_1, R$, and M. The challenge ciphertext is constructed as follows:

1. It obtains a cover $CV_R = \{(T_1, ID_1), \ldots, (T_l, ID_l)\}$ by running **PKCS.FindCover**$(\mathcal{T}_N, R)$. For $l+1 \leq j \leq L$, it sets $ID_j = dummy$.
2. For $1 \leq j \leq h$, it computes C_j by running **PE.Encrypt**$((\vec{x}_1, ID_j), M, PK')$.
3. For $h+1 \leq j \leq L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_0, ID_j), M, PK')$.
4. It outputs a ciphertext as $CT = \big(C_{\pi(1)}, \ldots, C_{\pi(L)}\big)$ where π is a random permutation.

Game $\mathbf{G}_{II,L}$. In this game $\mathbf{G}_{II,L}$, the challenge ciphertext is constructed as follows:

1. It obtains a cover $CV_R = \{(T_1, ID_1), \ldots, (T_l, ID_l)\}$ by running **PKCS.FindCover**$(\mathcal{T}_N, R)$. For $l+1 \leq j \leq L$, it sets $ID_j = dummy$.
2. For $1 \leq j \leq L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_1, ID_j), M, PK')$.
3. It outputs a ciphertext as $CT = \big(C_{\pi(1)}, \ldots, C_{\pi(L)}\big)$ where π is a random permutation.

Note that this game is the same as the original security game except that the challenge bit b is fixed to 1.

Let $\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,h}}$ be the advantage of $\mathcal{A}_{II}$ in $\mathbf{G}_{II,h}$. That is, $\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,h}} = |\Pr[b' = b | b = 0] - 1/2|$. We obtain $\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,L}} = |\Pr[b' = 0|b = 0] - 1/2| = |\Pr[b' = 0|b = 1] - 1/2|$ since $G_{II,L}$ is the same as the original security game where b is fixed to 1. In Lemma 5.5, we prove that it is hard to distinguish $\mathbf{G}_{II,h-1}$ from $\mathbf{G}_{II,h}$ if the underlying PE scheme is secure. Thus, we have that

$$\begin{aligned}\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,0}} - \mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,L}} &= \mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,0}} + \sum_{h=1}^{L-1} \left(\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,h}} - \mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,h}}\right) - \mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,L}} \\ &\leq \sum_{h=1}^{L} \left|\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,h-1}} - \mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,h}}\right| \leq 2L \cdot \mathsf{Adv}_{\mathcal{B}}^{PE,AH}(\lambda).\end{aligned}$$

Finally, we can obtain the inequality relation as

$$\begin{aligned}\mathsf{Adv}_{\mathcal{A}_{II}}^{RPE,wFH}(\lambda) &= \left|\Pr[b=0]\cdot\Pr[b'=b|b=0] + \Pr[b=1]\cdot\Pr[b'=b|b=1] - 1/2\right| \\ &= \left|1/2\cdot(\Pr[b'=0|b=0]-1/2) + 1/2\cdot((1-\Pr[b'=0|b=1]) - 1/2)\right| \\ &= 1/2\cdot\left|(\Pr[b'=0|b=0]-1/2) - (\Pr[b'=0|b=1]-1/2)\right| \\ &\leq 1/2\cdot\left|\mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,0}} - \mathsf{Adv}_{\mathcal{A}_{II}}^{G_{II,L}}\right| \leq L\cdot\mathsf{Adv}_{\mathcal{B}}^{PE,AH}(\lambda).\end{aligned}$$

This completes our proof. □

Lemma 5.5. *If there is a type-II adversary that distinguishes between $\mathbf{G}_{II,h-1}$ and $\mathbf{G}_{II,h}$ with non-negligible advantage, then there is a PPT algorithm $\mathcal{B}$ for the PE scheme such that $\left|Adv_{\mathcal{A}_{II}}^{G_{II,h-1}} - Adv_{\mathcal{A}_{II}}^{G_{II,h}}\right| \leq 2\cdot Adv_{\mathcal{B}}^{PE,AH}(\lambda)$.*

Proof. The basic idea of this proof is that a simulator can use the challenge oracle of PE to construct the challenge ciphertext component C_h of RPE since the only difference between $G_{II,h-1}$ and $G_{II,h}$ is the challenge ciphertext component C_h. Suppose there exists a type-II adversary $\mathcal{A}_{II}$ that distinguishes between $\mathbf{G}_{II,h-1}$ and $\mathbf{G}_{II,h}$ with non-negligible advantage ε. A simulator $\mathcal{B}$ that attacks the PE scheme is first given: a challenge public key PK'. Then $\mathcal{B}$ that interacts with $\mathcal{A}_{II}$ is described as follows:

Setup: $\mathcal{B}$ first sets $PK = (PK', N)$ and gives PK to $\mathcal{A}_{II}$.

Phase I: $\mathcal{A}_{II}$ adaptively requests a secret key for a vector $\vec{y}$ and a user index u. $\mathcal{B}$ proceeds the secret key query as follows:

1. It obtains a path value $PV_u = \{(T_0, ID_0), \ldots, (T_d, ID_d)\}$ by running **PKCS.GetPathValue**$(\mathcal{T}_N, u)$.
2. For $0 \leq j \leq d$, it sets a new predicate $f_{\vec{y},ID_j} = (\vec{y}, ID_j)$ and requests a secret key K_j for a new predicate $f_{\vec{y},ID_j}$ to the key generation oracle that simulates **PE.GenKey**.
3. It sets the secret key $SK_{\vec{y},u} = (K_0, \ldots, K_d)$ and gives this to $\mathcal{A}_{II}$.

Challenge: $\mathcal{A}_{II}$ outputs challenge vectors $\vec{x}_0, \vec{x}_1 \in \mathbb{Z}_p^n$, challenge sets $R_0, R_1 \subseteq \mathcal{U}$ such that $R_0 = R_1 = R$, and challenge messages $M_0, M_1 \in \mathcal{M}$ such that $M_0 = M_1 = M$ subject to the restrictions. It sets $b = 0$ and proceeds as follows:

1. It obtains a cover $CV_R = \{(T_1, ID_1), \ldots, (T_l, ID_l)\}$ by running **PKCS.FindCover**$(\mathcal{T}_N, R)$. For $l+1 \leq j \leq L$, it sets $ID_j = dummy$.
2. For $1 \leq j \leq h-1$, it computes C_j by running **PE.Encrypt**$((\vec{x}_1, ID_j), M, PK')$.

3. For $j = h$, it gives challenge attributes $x'_0 = (\vec{x}_0, ID_j), x'_1 = (\vec{x}_1, ID_j)$ and challenge messages $M'_0 = M, M'_1 = M$ to the challenge oracle that simulates **PE.Encrypt**, and receives C'. It sets $C_h = C'$.
4. For $h+1 \leq j \leq L$, it computes C_j by running **PE.Encrypt**$((\vec{x}_0, ID_j), M, PK')$.
5. It outputs a ciphertext as $CT = (C_{\pi(1)}, \ldots, C_{\pi(L)})$ where π is a random permutation.

Phase II: $\mathcal{A}_{II}$ may continue to request secret keys for additional vectors and user indexes subject to the same restrictions as before, and $\mathcal{B}$ gives the corresponding secrets keys to $\mathcal{A}_{II}$.

Output: Finally $\mathcal{A}_{II}$ outputs a bit b'. $\mathcal{B}$ sets $c' = b'$ and outputs c'.

To finish the proof, we show that the distribution of the simulation is correct. It is easy to check that the public key and the secret keys are correctly distributed. Let c be the challenger oracle's random bit of the PE scheme. If $c = 0$, then C' is the encryption of an attribute $x'_0 = (\vec{x}_0, ID_j)$ and a message M. If $c = 1$, then C' is the encryption of an attribute $x'_1 = (\vec{x}_1, ID_j)$ and a message M. Thus, the challenge ciphertext is the same as $\mathbf{G}_{II,h-1}$ if $c = 0$. Similarly, we obtain that the challenge ciphertext is the same as $\mathbf{G}_{II,h}$ if $c = 1$. Therefore, we have

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{B}}^{PE,AH}(\lambda) &= \Pr[c=0] \cdot \big|\Pr[c'=c|c=0] - 1/2\big| + \Pr[c=1] \cdot \big|\Pr[c'=c|c=1] - 1/2\big| \\
&= 1/2 \cdot \big|\Pr[b'=0|c=0] - 1/2\big| + 1/2 \cdot \big|\Pr[b'=1|c=1] - 1/2\big| \\
&= 1/2 \cdot \big|\Pr[b'=0|c=0] - 1/2\big| + 1/2 \cdot \big|(1 - \Pr[b'=0|c=1]) - 1/2\big| \\
&\geq 1/2 \cdot (\Pr[b'=0|c=0] - 1/2) - 1/2 \cdot (\Pr[b'=0|c=1] - 1/2) \\
&= 1/2 \cdot (\Pr[b'=0|b=0, c=0] - 1/2) - 1/2 \cdot (\Pr[b'=0|b=0, c=1] - 1/2) \\
&= 1/2 \cdot \mathsf{Adv}_{A_{II}}^{G_{II,h-1}} - 1/2 \cdot \mathsf{Adv}_{A_{II}}^{G_{II,h}}.
\end{aligned}$$

This completes our proof. □

5.4 Discussions

Efficiency. Let $N = 2^d$ be the number of users and n be the size of vectors for inner product operations in the RPE scheme. In this case, the underlying IPE scheme of the RPE scheme should support vectors of size $n+2$ for inner product operations. The secret key of the RPE scheme consists of $(d+1) = \log(N)$ secret keys of the IPE scheme, the ciphertext of the RPE scheme consists of $L = r\log(N/r) \approx r\log(N)$ ciphertexts of the IPE scheme, and the decryption algorithm of the RPE scheme requires $(d+1)L = r\log^2(N)$ decryption operations of the IPE scheme. For instance, if we use the IPE scheme with constant-size secret keys of Okamoto and Takashima [22], then the public key consists of $O(n)$ group elements, the secret key consists of $11\log(N)$ group elements, the ciphertext consists of $(5n+11)r\log(N)$ group elements, and the decryption algorithm requires $(n+2)r\log^2(N)$ exponentiations and $11r\log^2(N)$ pairing operations.

Efficient Decryption. The decryption algorithm of the RPE scheme requires $(d+1)L = r\log^2(N)$ decryption operations of the underlying IPE scheme. We can improve the efficiency of the decryption algorithm by using the technique of Barth et al. [5], redefined as anonymous hint systems by Libert et al. [18]. If we use the anonymous hint system of Barth et al., then we can reduce the decryption overhead from $r\log^2(N)$ decryption operations of the IPE scheme to $r\log^2(N)$ exponentiations and one decryption operation of the IPE scheme. For instance, if we use the IPE scheme with constant-size secret keys of Okamoto and Takashima [22], then the decryption algorithm just requires $r\log^2(N)$ exponentiations and 11 pairing operations.

6 Conclusion

In this paper, we revisited the notion of privacy preserving revocable predicate encryption (RPE) and its general constructions. We first introduced two security notions of RPE, namely the strongly full-hiding security and the weakly full-hiding security. Next, we proposed a general RPE construction with linear size ciphertexts that provides the strongly full-hiding security. To obtain a more efficient RPE scheme, we proposed another general RPE construction with sublinear size ciphertexts that provides the weakly full-hiding security. One advantage of our RPE constructions is that they can use the most efficient IPE scheme since they use the IPE scheme in a black-box way. Additionally, we can improve the efficiency of the decryption algorithm of the second RPE construction by using anonymous hint systems. One interesting problem is to propose a general RPE construction from any IPE scheme that has a ciphertext of sublinear size and provides the strongly full-hiding security.

Acknowledgements

Kwangsu Lee was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2012-H0301-12-3007) supervised by the NIPA (National IT Industry Promotion Agency). Seong Oun Hwang was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012002139).

References

[1] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2005.

[2] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In Daniele Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2010.

[3] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2011.

[4] Nuttapong Attrapadung and Hideki Imai. Conjunctive broadcast and attribute-based encryption. In Hovav Shacham and Brent Waters, editors, *Pairing*, volume 5671 of *Lecture Notes in Computer Science*, pages 248–265. Springer, 2009.

[5] Adam Barth, Dan Boneh, and Brent Waters. Privacy in encrypted content distribution using private broadcast encryption. In Giovanni Di Crescenzo and Aviel D. Rubin, editors, *Financial Cryptography*, volume 4107 of *Lecture Notes in Computer Science*, pages 52–64. Springer, 2006.

[6] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society, 2007.

[7] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security*, pages 417–426. ACM, 2008.

[8] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.

[9] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[10] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, 2011.

[11] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.

[12] Yevgeniy Dodis and Nelly Fazio. Public key broadcast encryption for stateless receivers. In Joan Feigenbaum, editor, *Digital Rights Management Workshop*, volume 2696 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2002.

[13] Nelly Fazio and Irippuge Milinda Perera. Outsider-anonymous broadcast encryption with sublinear ciphertexts. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 225–242. Springer, 2012.

[14] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.

[15] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.

[16] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.

[17] Kwangsu Lee and Dong Hoon Lee. Improved hidden vector encryption with short ciphertexts and tokens. *Des. Codes Cryptography*, 58(3):297–319, 2011.

[18] Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 206–224. Springer, 2012.

[19] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer, 2001.

[20] Juan Manuel González Nieto, Mark Manulis, and Dongdong Sun. Fully private revocable predicate encryption. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *ACISP*, volume 7372 of *Lecture Notes in Computer Science*, pages 350–363. Springer, 2012.

[21] Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. In Dongdai Lin, Gene Tsudik, and Xiaoyun Wang, editors, *CANS*, volume 7092 of *Lecture Notes in Computer Science*, pages 138–159. Springer, 2011.

[22] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 591–608. Springer, 2012.

[23] Jong Hwan Park. Inner-product encryption under standard assumptions. *Des. Codes Cryptography*, 58(3):235–257, 2011.

[24] Amit Sahai, Hakan Seyalioglu, and Brent Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 199–217. Springer, 2012.

[25] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.

[26] Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 560–578. Springer, 2008.