Conditional Attribute-Based Proxy Re-Encryption

Lisha Yao, Jian Weng*, Yi Liu, Junzuo Lai, and Bimei Wang

College of Cybersecurity, Jinan University, Guangzhou, China
yaolishaqh@gmail.com, cryptjweng@gmail.com, liuyi@jnu.edu.cn, laijunzuo@gmail.com, bimeiwang1@gmail.com

Abstract. Proxy re-encryption (PRE) is a cryptographic primitive that allows a semi-trusted proxy to transfer the decryption rights of ciphertexts in a secure and privacy-preserving manner. This versatile primitive has been extended to several powerful variants, leading to numerous applications, such as e-mail forwarding and content distribution. One such variant is attribute-based PRE (AB-PRE), which provides an *expressible access control mechanism* by allowing the proxy to switch the underlying policy of an attribute-based encryption (ABE) ciphertext. However, the function of AB-PRE is to convert the underlying policies of all ciphertexts indiscriminately, which lacks the flexibility of ciphertext transformation. Therefore, AB-PRE needs to support the property of conditional delegation. Among the other variants of PRE, there is a variant called conditional PRE (C-PRE), which allows *fine-grained delegations* by restricting the proxy to performing valid re-encryption only for a limited set of ciphertexts. Unfortunately, existing PRE schemes cannot *simultaneously* achieve expressible access control mechanisms and fine-grained delegations. Specifically, we require a PRE scheme, via which the proxy can convert the underlying policies of an ABE ciphertext only if this ciphertext is in the set of ciphertexts allowing the proxy to perform valid transformations.
To address this problem, we formalize the notion of conditional attribute-based PRE (CAB-PRE) in the honest re-encryption attacks (HRA) model, which is more robust and implies chosen-plaintext attacks (CPA) security, and propose the *first* CAB-PRE scheme. To construct such a scheme, we design as a building block, the *first* adaptively HRA-secure (ciphertext-policy) AB-PRE based on the learning with errors (LWE) problem. This scheme solves the open problem left by Susilo et al. [30] in ESORICS'21 about how to construct an HRA-secure (ciphertext-policy) AB-PRE scheme, and it should be of independent interest. Then, we introduce a well-matched conditional delegation mechanism for this AB-PRE scheme to derive our adaptively HRA-secure CAB-PRE scheme.

Keywords: Conditional attribute-based PRE · Honest re-encryption attacks · Learning with errors

1 Introduction

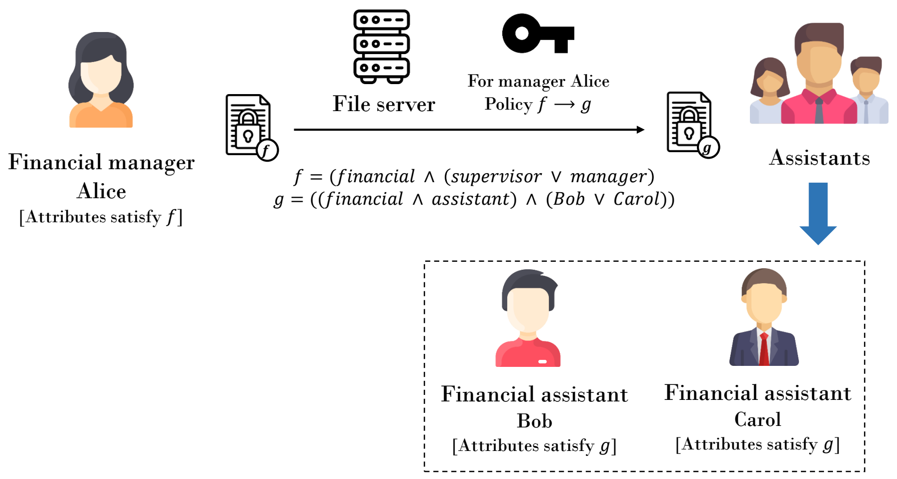
In 1998, Blaze, Bleumer, and Strauss introduced the notion of proxy re-encryption (PRE) [4] to enable the transfer of decryption rights between parties with the help of a semi-trusted proxy in a secure and privacy-preserving manner, is widely used in scenarios such as encrypted e-mail forwarding [25], online social networks [16], digital forensics evidence management [26], and CloudIoT platform [29,33]. More concretely, a PRE scheme provides a proxy holding a re-encryption key with the ability to transform a ciphertext that Alice (the delegator) can decrypt into one that can be decrypted by Bob (the delegatee). The proxy learns nothing about the plaintext and Alice's secret key in this transformation. In this paper, we restrict our attention to unidirectional, single-hop PRE [1].

Over the last two decades, PRE has been extended to several variants to support many practical application scenarios. One prominent variant is attribute-based proxy re-encryption (AB-PRE) [11, 13,19,20,22,30], extended from attribute-based encryption (ABE). AB-PRE scheme provides a more expressive access control mechanism than traditional PRE by allowing the proxy to switch the underlying policy for ABE ciphertexts. We use encrypted file processing as an example to illustrate the usage of AB-PRE. Imagine a company's financial manager, Alice, is to travel for a while. She delegates her assistants to process her regular files. The incoming files are encrypted under the policy

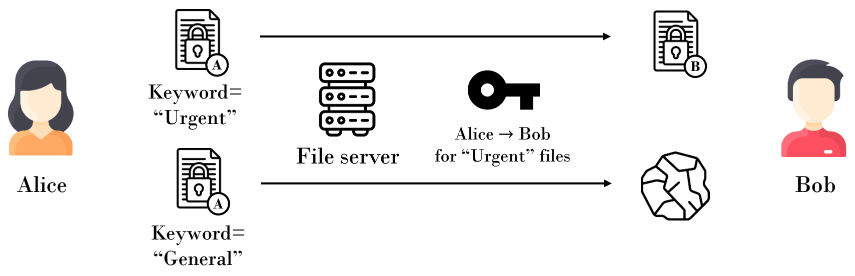
[1] Constructing bidirectional PRE schemes (*e.g.*, from Alice to Bob and vice versa) is straightforward, based on ones (*e.g.*, only from Alice to Bob). Single-hop PRE means that the delegatee Bob cannot delegate the decryption rights to others.

"f =(*financial* $\wedge$ (*supervisor* $\vee$ *manager*))", and Alice has the decryption rights. Traditional ABE schemes do not enable others to process Alice's files, following the security norm that Alice's secret key should not be shared with others. With an AB-PRE system (see Fig. 1(a)), Alice can give the file server (*i.e.*, the proxy) a re-encryption key. For an encrypted incoming file, the file server transforms it encrypted under the policy "f =(*financial* $\wedge$ (*supervisor* $\vee$ *manager*))" into one encrypted under the policy "g =((*financial* $\wedge$ *assistant*) $\wedge$ (*Bob* $\vee$ *Carol*))", such that the attributes of assistants satisfy g. Then Alice's assistants could read this file using their secret keys.

Although AB-PRE enables effective data sharing, it indiscriminately converts the underlying policies of all ciphertexts and lacks the flexibility of ciphertext transformation. Therefore, AB-PRE needs to support the property of conditional delegation. A notable PRE variant, called conditional PRE (C-PRE) [18,32], achieves fine-grained delegation by allowing the proxy to perform valid transformations only for ciphertexts satisfying the condition specified in the re-encryption key, and thus only a part of the ciphertexts can be transformed by the proxy. Again, let us take the above example of encrypted file processing to illustrate the usage of C-PRE. The incoming files are encrypted under Alice's public key and attach conditions, such as "*Urgent*" and "*General*". With a C-PRE system (see Fig. 1(b)), Alice can delegate her assistant Bob to process files only when its subject contains the keyword "*Urgent*" and then give a re-encryption key to the file server. For an encrypted incoming file, the file server transforms it into an encryption for Bob. Then Bob can read this file using his secret key.



(a) For AB-PRE, the file server holding a re-encryption key with respect to Alice can convert files encrypted under the policy f into ones encrypted under the policy g *if Alice has the decryption rights of the original files.*



(b) For C-PRE, the file server holding a re-encryption key with respect to Alice can *only* transfer the decryption rights of "*Urgent*" files from Alice to Bob.

Fig. 1. Illustrations of examples for AB-PRE and C-PRE.

In many application scenarios like distributed file systems, expressible access control mechanisms and fine-grained delegations are extremely important. As shown in Fig. 2, suppose that Alice is allowed to instruct her assistants to process files encrypted under the policy "f =(*financial* $\wedge$ (*supervisor* $\vee$ *manager*))" only when their subject contains the keyword "*Urgent*". For other files containing the keyword "*General*", Alice reads them by herself after returning to the company. However, no existing PRE schemes satisfy such requirements. To show further motivation, we consider the case that assumes Alice is a company's general manager and is enabled to delegate her assistants for different functions to handle different types of files when Alice is busy, *e.g.*, the assistants for administrative matters only handle files related to the keyword *administrative* and the assistants for business matters only handle files related to the keyword *business*.

Informally, we need a more powerful AB-PRE scheme whereby Alice has a fine-grained control over the delegation. In other words, using a proxy, Alice can flexibly assign her assistants the decryption rights based on the conditions attached to the re-encryption key. This paper calls such a desirable PRE scheme conditional attribute-based PRE (CAB-PRE) scheme. Fig. 2 illustrates the usage of CAB-PRE. The incoming files are encrypted under the policy "f =(*financial* $\wedge$ (*supervisor* $\vee$ *manager*))" and attach conditions, such as "*Urgent*" and "*General*", and Alice has the decryption rights. With a CAB-PRE system, Alice can give the file server a re-encryption key. For an encrypted incoming file, the file server converts it with the keyword "*Urgent*" into one encrypted under the policy "g =((*financial* $\wedge$ *assistant*) $\wedge$ (*Bob* $\vee$ *Carol*))", such that the attributes of assistants satisfy g. Then Alice's assistants could read this file using their secret keys.

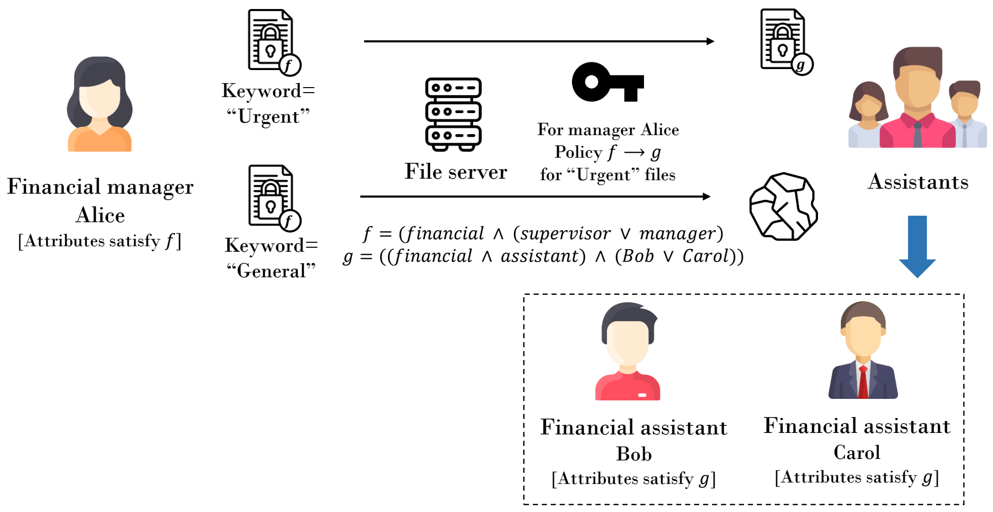


Fig. 2. For CAB-PRE, the file server holding a re-encryption key with respect to Alice can *only* convert "*Urgent*" files encrypted under the policy f into ones encrypted under the policy g *if Alice has the decryption rights of the original files.*

In this paper, we aim to formalize the notion of CAB-PRE and propose a CAB-PRE scheme. For the formalization and construction, we also consider the following additional factors.

Security Model It has been pointed out that chosen-plaintext attack (CPA) security is inadequate to preserve the privacy of the delegator's secret key from a single honestly re-encrypted ciphertext for PRE schemes. The security notion under honest re-encryption attacks (HRA) can better capture this security requirement of PRE (see [9, 12] for more). Hence, we focus on the HRA security model to formalize and design our CAB-PRE scheme.

Hardness Assumption With the fast development of quantum computers, we now prefer to design schemes that are also secure in the post-quantum world. Thus, we intend to design a CAB-PRE scheme that is resistant to attack by quantum computers.

Predicate Expressibility An expressive class of predicates is desirable for the underlying AB-PRE. Meanwhile, we also expect the delegation mechanism to be fine-grained, *i.e.*, to support an expressive class of predicates.

1.1 Our Results

In this paper, we formalize the notion of CAB-PRE in the HRA security model and propose the *first* adaptively HRA-secure CAB-PRE scheme based on the learning with errors (LWE) problem [28], a well-known post-quantum cryptographic assumption. The underlying policy of our CAB-PRE scheme supports t-conjunctive normal form (t-CNF) predicates, including NP-verification policies, bit-fixing policies, and t-threshold policies. The delegation mechanism of the CAB-PRE scheme supports inner-product predicates. The main contributions of this paper are summarized as follows.

1. Considering the need for expressible access control mechanisms and fine-grained delegations in many scenarios, we formalize the notion of CAB-PRE in the HRA security model.
2. To construct our CAB-PRE scheme, we design as a building block, the *first* adaptively HRA-secure AB-PRE scheme for t-CNF predicates based on the LWE problem. We note that this scheme also serves as the solution to the open problem introduced by Susilo et al. [30] in ESORICS'21 about how to construct an HRA-secure (ciphertext-policy) AB-PRE scheme, and it should be of independent interest. We further compare related lattice-based AB-PRE schemes and our scheme in Table 1. The results show that our scheme achieves adaptive HRA security while maintaining the same order of magnitude in parameters, keys, and ciphertext sizes as other relevant schemes without incurring more computational overhead.
3. We introduce a well-matched conditional delegation mechanism tailored to inner-product predicates and integrate it into our AB-PRE scheme to obtain the *first* CAB-PRE scheme, which is adaptively HRA secure.

Table 1. Comparison between related lattice-based AB-PRE schemes and our scheme

Scheme / Feature	Li et al. [19]	Luo et al. [22]	Susilo et al. [30]	Our scheme
Policy	Ciphertext-Policy	Key-Policy	Key-Policy	Ciphertext-Policy
Security	Selective CPA	Selective CPA	Selective HRA	Adaptive HRA
Public parameters size	$O(2l \cdot n^2 \log^2 q)$	$O((l+2) \cdot n^2 \log^2 q)$	$O((l+3) \cdot n^2 \log^2 q)$	$O((\lambda+1) \cdot n^2 \log^2 q)$
Master secret key size	$O(2l \cdot n^2 \log^3 q)$	$O(n^2 \log^3 q)$	$O(n^2 \log^3 q)$	$O(n^2 \log^3 q)$
Secret key size	$O(l \cdot n \log^2 q)$	$O(4n^2 \log^3 q)$	$O(2n^2 \log^3 q)$	$O(4n^2 \log^3 q)$
Ciphertext size	$O(2l \cdot n \log^2 q)$	$O((l+2) \cdot n \log^2 q)$	$O((l+2) \cdot n \log^2 q)$	$O((l_f+1) \cdot n \log^2 q)$
Re-encryption key size	$O(2l \cdot n^2 \log^3 q)$	$O(4n^2 \log^3 q)$	$O((l+2) \cdot n^2 \log^4 q)$	$O((l_g+1) \cdot n^2 \log^4 q)$
Re-encryption ciphertext size	$O(2l \cdot n \log^2 q)$	$O(3n \log^2 q)$	$O((l+2) \cdot n \log^2 q)$	$O((l_f+1) \cdot n \log^2 q)$

l is the size of attribute vectors;
λ is security parameter;
l_f and l_g are the size of constrained key associated with functions f and g, respectively.

1.2 Organization

This paper is organized as follows. We provide an overview of related cryptographic techniques and background on lattices in Section 2. Next, we define the syntax and security model for CAB-PRE in Section 3. Our adaptively HRA-secure AB-PRE scheme and its security proof are introduced in Section 4. Then we give an adaptively HRA-secure CAB-PRE based on the above AB-PRE in Section 5. Finally, we conclude this paper with future work in Section 6.

2 Preliminaries

In this section, we introduce some cryptography preliminaries and background on lattices to help the reader better capture the following chapters.

2.1 Constrained PRF, Conforming cPRF and t-CNF Predicates

We review two key concepts: constrained pseudorandom function (cPRF) and conforming cPRF, which are significant components in our constructions. To facilitate comprehension, we first introduce the concept of cPRF and augment cPRF with the properties of gradual evaluation and key simulation to obtain conforming cPRF.

Definition 1 (Constrained PRF [10,31]). *Let $\mathcal{F}$ be a family of functions with domain $\{0,1\}^l$ and range $\{0,1\}$. A constrained pseudorandom function (cPRF) for $\mathcal{F}$ is defined by a tuple of probabilistic polynomial-time (PPT) algorithms $\Pi_{\mathsf{cPRF}} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Constrain}, \mathsf{ConstrainEval})$ where:*

- $\mathsf{Setup}(1^\lambda) \to (pp, msk)$: *The setup algorithm takes as input the security parameter 1^λ, outputs a public parameter pp and a master secret key msk.*
- $\mathsf{Eval}(msk, x) \to y$: *The evaluation is a deterministic algorithm which takes as input the master secret key msk and a bit-string $x \in \{0,1\}^l$, outputs $y \in \{0,1\}^k$.*
- $\mathsf{Constrain}(msk, f) \to sk_f$: *The constrained key generation takes as input the master secret key msk and a function $f \in \mathcal{F}$ specifying the constraint, outputs a constrained key sk_f.*
- $\mathsf{ConstrainEval}(sk_f, x) \to y'$: *The constrained evaluation is a deterministic algorithm which takes as input a constrained key sk_f and a bit-string $x \in \{0,1\}^l$, outputs $y' \in \{0,1\}^k$.*

Correctness. *We say a cPRF scheme Π_{cPRF} is correct if for all $f \in \mathcal{F}$ and $x \in \{0,1\}^l$ such that $f(x) = 1$, we have $\mathsf{Eval}(msk, x) = \mathsf{ConstrainEval}(sk_f, x)$, where $(pp, msk) \leftarrow \mathsf{Setup}(1^\lambda)$ and $sk_f \leftarrow \mathsf{Constrain}(msk, f)$.*

Pseudorandomness. *The single-key adaptive security of a cPRF is defined formally by the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:*

- **Setup** : *At the beginning of the game, the challenger $\mathcal{C}$ prepares $(pp, msk) \leftarrow \mathsf{Setup}(1^\lambda)$ and sends pp to $\mathcal{A}$.*
- **Phase 1** : *$\mathcal{A}$ can adaptively make two types of queries:*
  - *Evaluation Queries: Upon a query $x \in \{0,1\}^l$, the challenger evaluates $y \leftarrow \mathsf{Eval}(msk, x)$ and returns y to $\mathcal{A}$.*
  - *Constrained Key Queries: This oracle can only be queried once. Upon a query $f \in \mathcal{F}$, the challenger computes $sk_f \leftarrow \mathsf{Constrain}(msk, f)$ and returns sk_f to $\mathcal{A}$.*
- **Challenge** : *$\mathcal{A}$ chooses a target bit-string $x^* \in \{0,1\}^l$. The challenger flips a coin $b \overset{\$}{\leftarrow} \{0,1\}$. If $b = 1$, it evaluates $y^* \leftarrow \mathsf{Eval}(msk, x^*)$. Otherwise, it samples $y^* \overset{\$}{\leftarrow} \{0,1\}^k$. Finally, $\mathcal{C}$ returns y^* to $\mathcal{A}$.*
- **Phase 2** : *$\mathcal{A}$ continues to make queries as same as Phase 1.*
- **Guess** : *Eventually, $\mathcal{A}$ outputs b' as a guess for b.*

The adversary $\mathcal{A}$ wins the game if (1) $b' = b$, (2) x^* cannot be queried in the evaluation oracle and (3) all of the key queries for f satisfy $f(x^*) = 0$. The advantage that $\mathcal{A}$ wins in the security game is at most $1/2 + \mathsf{negl}(\lambda)$.

Definition 2 (Conforming cPRF [31]). *We call a cPRF scheme is conforming, besides correctness and pseudorandomness defined above, if the following properties hold.*

Gradual Evaluation. *Let* $\mathsf{Constrain}$ *(in addition to* $\mathsf{Eval}, \mathsf{ConstrainEval}$*) algorithm be deterministic. Fixing pp from* $\mathsf{Setup}(1^\lambda)$*, for any $f \in \mathcal{F}$ and $x \in \{0,1\}^l$ such that $f(x) = 1$, define the following circuits:*

- $U_{\sigma \to x} : \{0,1\}^\lambda \to \{0,1\}^k$ *takes as input msk and computes* $\mathsf{Eval}(msk, x)$.
- $U_{\sigma \to f} : \{0,1\}^\lambda \to \{0,1\}^{l_f}$ *takes as input msk and computes* $\mathsf{Constrain}(msk, f)$.
- $U_{f \to x} : \{0,1\}^{l_f} \to \{0,1\}^k$ *takes as input sk_f and computes* $\mathsf{ConstrainEval}(sk_f, x)$.

Note that the circuit $U_{\sigma \to x}$ is the same as the efficient sub-circuit of $U_{f \to x} \circ U_{\sigma \to f}$.

Key Simulation. *We require a PPT algorithm* $\mathsf{KeySim}(pp, f) \to sk_f$ *such that any adversary $\mathcal{A}$ has at most $1/2 + \mathsf{negl}(\lambda)$ advantage of winning the following game against a challenger $\mathcal{C}$.*

- **Setup** : *$\mathcal{C}$ generates $(pp, msk) \leftarrow \mathsf{Setup}(1^\lambda)$ and sends pp to $\mathcal{A}$.*
- **Phase 1** : *$\mathcal{A}$ makes evaluation queries for polynomial times. For a bit-string $x \in \{0,1\}^l$, $\mathcal{C}$ returns $y \leftarrow \mathsf{Eval}(msk, x)$.*
- **Challenge** : *For a challenge constraint $f^* \in \mathcal{F}$, $\mathcal{C}$ samples $b \overset{\$}{\leftarrow} \{0,1\}$ and returns $sk_{f^*} \leftarrow$* $\mathsf{Constrain}(msk, f^*)$ *if $b = 0$, otherwise it returns $sk_{f^*} \leftarrow \mathsf{KeySim}(pp, f^*)$.*

– **Phase 2** : *Same as the queries of Phase 1.*
– **Guess** : *$\mathcal{A}$ outputs a bit b'.*
$\mathcal{A}$ wins the game if (1) $b' = b$ and (2) all of the evaluation queries for x satisfy $f^*(x) = 0$.

In this work, we utilize t-conjunctive normal form (t-CNF) predicates to specify the access policy, *i.e.*, the constraint f in cPRF, where each clause exhibits constant locality.

Definition 3 (t-CNF Predicates [10, 31]). *A t-CNF predicate $f : \{0,1\}^l \to \{0,1\}$ such that $t \leq l$ is a set of clauses $f = \{(T_i, f_i)\}_i$, where for all i, $T_i \subseteq [l]$, $|T_i| = t$ and $f_i : \{0,1\}^t \to \{0,1\}$. For all $x \in \{0,1\}^l$, a t-CNF predicate $f(x)$ is computed as*

$$f(x) = \bigwedge_i f_i(x_{T_i})$$

where $x_{T_i} \in \{0,1\}^t$ is the bit string consisting of the bits of x in the indices of T_i. At last, a family of t-CNF predicates $\mathcal{F}$ is the set of t-CNF predicates with input length l.

Tsabary [31] shows that the PRF [15] is a conforming cPRF for prefix policies. Moreover, he gives a conforming cPRF construction for t-CNF policies, which is inspired by the [10] construction of bit-fixing cPRF for a constant number of keys.

2.2 Lattices, Discrete Gaussian, Bounded Distributions

Notations. Bold symbols are used to represent matrices or vectors, while regular lowercase letters are used for individual elements. Let $(\cdot\|\cdot)$ (resp. $(\cdot;\cdot)$) denote the horizontal concatenation (resp. vertical concatenation) of vectors or matrices. For a distribution on set X, we use $x \overset{\$}{\leftarrow} X$ to denote a random sample from X. The symbols $\wedge$ and $\vee$ denote the "AND" gate and "OR" gate, respectively. Let $\models$ (resp. $\not\models$) denote the satisfied relation (resp. dissatisfied relation) between two conditions. We use $\overset{c}{\approx}$ as the abbreviation for computationally indistinguishable.

Matrix Norm. For a vector $\mathbf{u}$, let $\|\mathbf{u}\|$ denote its l_2 norm. For a matrix $\mathbf{R} \in \mathbb{Z}^{n\times m}$, let $\widetilde{\mathbf{R}}$ be the Gram-Schmidt orthogonalization of $\mathbf{R}$. We define the following matrix norms:

– $\|\mathbf{R}\|$ denotes the l_2 length of the longest column of $\mathbf{R}$.
– $\|\mathbf{R}\|_\infty$ denotes the maximum element in $\mathbf{R}$.

Note that $\|\mathbf{R}\|_\infty \leq \|\mathbf{R}\| \leq nm\|\mathbf{R}\|_\infty$ and that $\|\mathbf{RS}\|_\infty \leq m\|\mathbf{R}\|_\infty\|\mathbf{S}\|_\infty$.

Lattices. In this work, two kinds of integer lattices are used. For a prime q, given $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{u} \in \mathbb{Z}_q^n$, denote:

$$\Lambda_q^\perp(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{Av} = \mathbf{0} \pmod q\},$$
$$\Lambda_q^{\mathbf{u}}(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{Av} = \mathbf{u} \pmod q\}.$$

Observe that if $\mathbf{t} \in \Lambda_q^{\mathbf{u}}(\mathbf{A})$ then $\Lambda_q^{\mathbf{u}}(\mathbf{A}) = \Lambda_q^\perp(\mathbf{A}) + \mathbf{t}$ and hence $\Lambda_q^{\mathbf{u}}(\mathbf{A})$ is a shift of $\Lambda_q^\perp(\mathbf{A})$.

Gadget Matrix. Let $n, q \in \mathbb{Z}$, $\mathbf{g} = (1, 2, 4, \cdots, 2^{\lceil\log q\rceil - 1})^T \in \mathbb{Z}_q^{\lceil\log q\rceil}$ and $m = n\lceil\log q\rceil$. The gadget matrix is defined as $\mathbf{G} = \mathbf{g}^T \otimes \mathbf{I}_n \in \mathbb{Z}_q^{n\times m}$, which denotes a tensor product of a vector $\mathbf{g}$ and an identity matrix $\mathbf{I}_n$ such that the lattice $\Lambda_q^\perp(\mathbf{G})$ has a public known basis $\mathbf{T_G}$ with $\|\widetilde{\mathbf{T_G}}\| \leq \sqrt{5}$.

Discrete Gaussian [1]. Let L be a subset of $\mathbb{Z}^m$. For any vector $\mathbf{c} \in \mathbb{R}^m$ and any positive parameter $\sigma \in \mathbb{R}$, define:

$$\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\frac{\|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2}) \text{ and } \rho_{\sigma,\mathbf{c}}(L) = \sum_{\mathbf{x}\in L}\rho_{\sigma,\mathbf{c}}(\mathbf{x}).$$

A discrete Gaussian distribution on L with center $\mathbf{c}$ and parameter σ is

$$\forall\ \mathbf{y} \in L, \mathcal{D}_{L,\sigma,\mathbf{c}}(\mathbf{y}) = \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(L)}.$$

The distribution $\mathcal{D}_{L,\sigma}$ ($\mathbf{c} = \mathbf{0}$ when omitted) is most often defined over a lattice $L = \Lambda_q^\perp(\mathbf{A})$ for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ or over a coset $L = \Lambda_q^\perp(\mathbf{A}) + \mathbf{t}$, where $\mathbf{t} \in \mathbb{Z}^m$.

Tailcut. The "tailcut" property of the discrete Gaussian is a crucial characteristic that enables the bounding of parameters.

Definition 4 ([1,24]). *Let $q \geq 2$, $m > n$ and $\mathbf{A}$ be a matrix in $\mathbb{Z}_q^{n\times m}$. Let $\mathbf{T_A}$ be a basis for $\Lambda_q^{\perp}(\mathbf{A})$ and $\tau \geq \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log m})$. For all $\mathbf{u} \in \mathbb{Z}_q^n$, we have*

$$\Pr[\mathbf{x} \xleftarrow{\$} \mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\tau} : \|\mathbf{x}\| > \tau\sqrt{m}] \leqslant \mathsf{negl}(\lambda).$$

Bounded Distributions. The following properties can help us set the parameters appropriately.

Definition 5 ([7,31]). *A distribution χ supported over $\mathbb{Z}$ is (B,ϵ)-bounded, if we have* $\Pr[x \xleftarrow{\$} \chi : |x| > B] < \epsilon$.

Definition 6 ([7, 31]). *A distribution $\tilde{\chi}$ supported over $\mathbb{Z}$ is (B,ϵ)-swallowing if for all $y \in [-B,B] \cap \mathbb{Z}$, we have that $\tilde{\chi}$ and $y + \tilde{\chi}$ are within ϵ statistical distance.*

2.3 Lattice Algorithms, Lattice Evaluation and LWE

Lattice Algorithms. In this paper, we will use several lattice algorithms, which are enumerated in the following lemmas:

Lemma 1 (TrapGen [23]). *Let $n,m,q > 0$ be integers with $m \geq O(n\log q)$. A PPT algorithm* $\mathsf{TrapGen}(1^n, m, q)$ *outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a full-rank matrix $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$, where $\mathbf{T_A}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$ and $\|\widetilde{\mathbf{T_A}}\| = O(\sqrt{n\log q})$. The distribution of $\mathbf{A}$ is $2^{-\Omega(n)}$-close to uniform.*

Lemma 2 (SamplePre [14]). *Let $q \geq 2$, $m > n$. A PPT algorithm* $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}, \tau)$ *that, given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a basis $\mathbf{T_A}$ for $\Lambda_q^{\perp}(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ and a Gaussian parameter $\tau \geq \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log m})$, outputs a vector $\mathbf{e} \in \mathbb{Z}^m$ sampled from a distribution $2^{-\Omega(n)}$-close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\tau}$.*

Lemma 3 (SampleLeft [1]). *Let $q > 2$, $m > n, m_1 > 0$. A PPT algorithm* $\mathsf{SampleLeft}(\mathbf{A}, \mathbf{T_A}, \mathbf{B}, \mathbf{u}, \tau)$ *that, given matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{B} \in \mathbb{Z}_q^{n\times m_1}$, a basis $\mathbf{T_A}$ for $\Lambda_q^{\perp}(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ and a Gaussian parameter τ, where $\tau \geq \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log(m + m_1)})$, outputs a vector $\mathbf{e} \in \mathbb{Z}^{m+m_1}$ sampled from a distribution $2^{-\Omega(n)}$-close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}\|\mathbf{B}),\tau}$.*

Lemma 4 (SampleRight [1]). *Let $q > 2$, $m > n$. A PPT algorithm* $\mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}, \mathbf{T_G}, \mathbf{u}, \tau)$ *that, given matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times k}, \mathbf{G} \in \mathbb{Z}_q^{n\times m}, \mathbf{R} \in \mathbb{Z}^{k\times m}$, a basis $\mathbf{T_G}$ for $\Lambda_q^{\perp}(\mathbf{G})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ and a Gaussian parameter $\tau \geq \|\widetilde{\mathbf{T_G}}\| \cdot s_R \cdot \omega(\sqrt{\log m})$ (where $s_R := \|\mathbf{R}\|$), outputs a vector $\mathbf{e} \in \mathbb{Z}^{m+k}$ sampled from a distribution $2^{-\Omega(n)}$-close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}\|\mathbf{AR}+\mathbf{G}),\tau}$.*

Lemma 5 (ExtendRight [5,8]). *Let $n, m_1, m_2, q > 0$ be integers with q prime. A PPT algorithm* $\mathsf{ExtendRight}(\mathbf{A}, \mathbf{T_A}, \mathbf{B})$ *that, given matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m_1}, \mathbf{B} \in \mathbb{Z}_q^{n\times m_2}$ and a basis $\mathbf{T_A}$ for $\Lambda_q^{\perp}(\mathbf{A})$, outputs a basis $\mathbf{T}_{(\mathbf{A}\|\mathbf{B})}$ for $\Lambda_q^{\perp}(\mathbf{A}\|\mathbf{B})$ such that $\|\widetilde{\mathbf{T_A}}\| = \|\widetilde{\mathbf{T}_{(\mathbf{A}\|\mathbf{B})}}\|$.*

Lemma 6 (ExtendLeft [5,8]). *Let $n, m, q > 0$ be integers with q prime. A PPT algorithm* $\mathsf{ExtendLeft}$ $(\mathbf{A}, \mathbf{G}, \mathbf{T_G}, \mathbf{R})$ *that, given matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times k}, \mathbf{G} \in \mathbb{Z}_q^{n\times m}, \mathbf{R} \in \mathbb{Z}^{k\times m}$, a basis $\mathbf{T_G}$ for $\Lambda_q^{\perp}(\mathbf{G})$, outputs a basis $\mathbf{T_H}$ for $\Lambda_q^{\perp}(\mathbf{H})$, where $\mathbf{H} = (\mathbf{A}\|\mathbf{AR} + \mathbf{G})$, such that $\|\widetilde{\mathbf{T_H}}\| \leq \|\widetilde{\mathbf{T_G}}\|(1 + \|\mathbf{R}\|)$.*

Lemma 7 (RandBasis [5,8]). *Let $m, q \geq 2$ be integers with q prime. A PPT algorithm* $\mathsf{RandBasis}$ $(\mathbf{A}, \mathbf{T_A}, \tau)$ *that, given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a basis $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ for $\Lambda_q^{\perp}(\mathbf{A})$ and a Gaussian parameter $\tau = \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log m})$, outputs a basis $\mathbf{T}'_{\mathbf{A}}$ for $\Lambda_q^{\perp}(\mathbf{A})$ sampled from a distribution that is statistically close to $\mathcal{D}^m_{\Lambda_q^{\perp}(\mathbf{A}),\sigma}$. Note that $\|\widetilde{\mathbf{T}'_{\mathbf{A}}}\| < \tau\sqrt{m}$ with all but negligible probability.*

Next, we introduce the abstraction form of lattice evaluation as extracted in [31].

Theorem 1 (Lattice Evaluation [31]). *Let $n, q, l, k \in \mathbb{N}$ and $m = n\lceil \log q \rceil$, there exist two deterministic algorithms called* **EvalF** *and* **EvalFX**, *respectively. For any depth d boolean circuit $f : \{0,1\}^l \to \{0,1\}^k$, for every $x \in \{0,1\}^l$ and for any matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times ml}$, the outputs $\mathbf{H} \leftarrow \mathbf{EvalF}(f, \mathbf{A})$ and $\hat{\mathbf{H}} \leftarrow \mathbf{EvalFX}(f, x, \mathbf{A})$ are both in $\mathbb{Z}^{ml\times mk}$ and it holds that $\|\mathbf{H}\|_\infty$, $\|\hat{\mathbf{H}}\|_\infty \leqslant (2m)^d$ and*

$$[\mathbf{A} - x \otimes \mathbf{G}]\hat{\mathbf{H}} = \mathbf{A}\mathbf{H} - f(x) \otimes \mathbf{G} \pmod q.$$

Moreover, for any pair of circuits f: $\{0,1\}^l \to \{0,1\}^k$, g: $\{0,1\}^k \to \{0,1\}^t$ and for any matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times ml}$, the outputs $\mathbf{H}_f \leftarrow \mathbf{EvalF}(f, \mathbf{A})$, $\mathbf{H}_g \leftarrow \mathbf{EvalF}(g, \mathbf{A}\mathbf{H}_f)$ and $\mathbf{H}_{g\circ f} \leftarrow \mathbf{EvalF}(g \circ f, \mathbf{A})$ satisfy $\mathbf{H}_f\mathbf{H}_g = \mathbf{H}_{g\circ f}$.

Learning With Errors. For the learning with errors (LWE) problem [28], we will use its decisional version, Hermite normal form (HNF) and lossy mode, in this work.

Definition 7 (Decisional LWE (DLWE) and Its HNF [2,28]). *Let λ be the security parameter, $n = n(\lambda)$ and $q = q(\lambda)$ be integers and let $\chi = \chi(\lambda)$ be a probability distribution over $\mathbb{Z}$. The DLWE$_{n,q,\chi}$ problem states that for all $m = \mathsf{poly}(n)$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \xleftarrow{\$} \chi^m$, and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$, it holds that*

$$(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e}) \quad \text{and} \quad (\mathbf{A}, \mathbf{u})$$

are computationally indistinguishable. The form of HNF-LWE is identical to the above except for $\mathbf{s} \xleftarrow{\$} \chi^n$.

Corollary 1 ([27,28,31]). *For all $\epsilon > 0$, there exist functions $q = q(n) \leq 2^n$, $\chi = \chi(n)$ such that χ is B-bounded for some $B = B(n)$, $q/B \leq 2^{n^\epsilon}$ and such that DLWE$_{n,q,\chi}$ is at least as hard as the classical hardness of GapSVP_γ and the quantum hardness of SIVP_γ for $\gamma = 2^{\Omega(n^\epsilon)}$.*

Definition 8 (Lossy Mode for LWE [17]). *The LWE instance $(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m$ uniquely determines a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$, provided the matrix $\mathbf{A}$ is drawn uniformly from $\mathbb{Z}_q^{n\times m}$ with sufficiently large m. However, suppose sample $\mathbf{A}$ from a specialized distribution that is computationally indistinguishable from the uniform distribution over $\mathbb{Z}_q^{n\times m}$. In that case, the information leakage regarding the secret $\mathbf{s}$ by the pair $(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e})$ is negligible. Consequently, this variant of the LWE problem, which does not reveal significant information about the secret vector, is commonly referred to as the "lossy mode".*

2.4 Vector Decomposition and Generalized Leftover Hash Lemma

Key Switching. We describe some subroutines of the key switching procedure proposed in [6] to control error growth in our scheme.

Definition 9 (Vector Decomposition). *We define the vector decomposition functions mapping vectors to a higher dimension as below:*

- $\mathsf{BD}(\mathbf{v})$: *A deterministic function that given a vector $\mathbf{v} \in \mathbb{Z}_q^n$, let $v_i^T \in \{0,1\}^n$ [2] be such that $\mathbf{v} = \sum_{i=0}^{\lceil \log q \rceil - 1} 2^i v_i$, outputs a vector $\widetilde{v}^T \in \{0,1\}^{n\lceil \log q \rceil}$, where $\widetilde{v} = (v_0; \cdots; v_{\lceil \log q \rceil - 1})$. If there is a row vector $\mathbf{v} \in \mathbb{Z}_q^{1\times n}$, we compute $\mathsf{BD}(\mathbf{v})$ as : let $\mathbf{v}^T \in \mathbb{Z}_q^n$, evaluate $\mathsf{BD}(\mathbf{v}^T)$, then we have $\mathsf{BD}(\mathbf{v}) = (\mathsf{BD}(\mathbf{v}^T))^T$.*

- $\mathsf{P2}(\mathbf{x})$: *A deterministic function that given a vector $\mathbf{x} \in \mathbb{Z}_q^n$, outputs a vector $\bar{\mathbf{x}} \in \mathbb{Z}_q^{n\lceil \log q \rceil}$, where $\bar{\mathbf{x}} = (\mathbf{x}; 2\mathbf{x}; \cdots; 2^{\lceil \log q \rceil - 1}\mathbf{x})$.*

- *For vectors $\mathbf{v} \in \mathbb{Z}_q^{1\times n}$ and $\mathbf{x} \in \mathbb{Z}_q^n$, it holds that $\mathsf{BD}(\mathbf{v}) \cdot \mathsf{P2}(\mathbf{x}) = \mathbf{v} \cdot \mathbf{x} \bmod q$.*

[2] By default, a bit string is considered as a row vector.

Randomness Extraction. We introduce a generalization version of the leftover hash lemma from [1].

Definition 10 (Generalized Leftover Hash Lemma [1]). *Suppose that $m > (n+1)\log q + \omega(\log n)$ and that $q > 2$ is prime. Let $\mathbf{S} \xleftarrow{\$} \{1,-1\}^{m\times k}$, where $k = k(n)$. Choose matrices $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n\times k}$. Then, for all vectors $\mathbf{e} \in \mathbb{Z}_q^m$, the distribution $(\mathbf{A}, \mathbf{AS}, \mathbf{S}^T\mathbf{e})$ is statistically close to the distribution $(\mathbf{A}, \mathbf{B}, \mathbf{S}^T\mathbf{e})$.*

Obviously, the two distributions $(\mathbf{A}, \mathbf{AS})$ and $(\mathbf{A}, \mathbf{B})$ are still statistically close in the case of without revealing some small amount of information of $\mathbf{S}$ (*i.e.*, $\mathbf{S}^T\mathbf{e}$).

3 Definition of Conditional Attribute-Based Proxy Re-encryption

We formally define conditional attribute-based proxy re-encryption (CAB-PRE) in the HRA security model. Specifically, we describe the syntax of unidirectional, single-hop CAB-PRE for ciphertext policy.

3.1 Conditional Attribute-Based Proxy Re-encryption

We note that combining C-PRE and AB-PRE constructions cannot directly yield a viable CAB-PRE scheme because their underlying encryption and re-encryption have different structures, making it hard to guarantee correct decryptions for transformed ciphertexts.

We formalize CAB-PRE with the identical syntax as the general AB-PRE. Differently, we furnish two levels of encryption algorithms to distinguish between pre- and post-delegation ciphertexts to improve their comprehensibility, in which the second level with specified conditions. When performing a re-encryption operation, the proxy converts the delegator's ciphertext generated by the second-level encryption algorithm into the delegatee's ciphertext produced by the first-level ones. Both encryption algorithms use the same decryption procedure to minimize the users' local computational burden. Therefore, the CAB-PRE scheme consists of seven algorithms in the following way.

Definition 11 (Conditional Attribute-Based Proxy Re-Encryption for Ciphertext Policy). *Let $\mathcal{F} : \{0,1\}^l \to \{0,1\}$ be a function class. A conditional attribute-based proxy re-encryption scheme CAB-PRE for policies in $\mathcal{F}$ is a tuple of PPT algorithms $\Pi_{\mathsf{CAB-PRE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}_1, \mathsf{Enc}_2, \mathsf{Dec}, \mathsf{ReKeyGen}, \mathsf{ReEnc})$.*

$\mathsf{Setup}(1^\lambda) \to (pp, msk)$. *On input the security parameter 1^λ, the setup algorithm outputs the public parameters pp along with a master secret key msk.*

$\mathsf{KeyGen}(msk, x) \to sk_x$**.** *On input a master secret key msk and an attribute string $x \in \{0,1\}^l$, the key generation algorithm outputs a secret key sk_x.*

$\mathsf{Enc}_1(f, \mu) \to ct_f^1$. *On input a policy $f \in \mathcal{F}$ and a plaintext $\mu \in \{0,1\}$, the first-level encryption algorithm outputs a first-level ciphertext ct_f^1 associated with policy f.*

$\mathsf{Enc}_2(f, w, \mu) \to ct_f^2$. *On input a policy $f \in \mathcal{F}$, a condition w and a plaintext $\mu \in \{0,1\}$, the second-level encryption algorithm outputs a second-level ciphertext ct_f^2 associated with policy f. Note that in single-hop, one could distinguish first-level ciphertexts from second-level ciphertexts.*

$\mathsf{Dec}(sk_x, ct_f \in \{ct_f^1, ct_f^2\}) \to \mu/\perp$. *On input a secret key sk_x and a ciphertext $ct_f \in \{ct_f^1, ct_f^2\}$, the decryption algorithm outputs a bit $\mu \in \{0,1\}$ if $f(x) = 1$ [3], else outputs an error symbol $\perp$.*

$\mathsf{ReKeyGen}(sk_x, g, w') \to rk_{x\to g}^{w'}$. *Given a secret key sk_x, a policy $g \in \mathcal{F}$, and a condition w', this algorithm outputs a re-encryption key $rk_{x\to g}^{w'}$.*

$\mathsf{ReEnc}(rk_{x\to g}^{w'}, ct_f^2) \to ct_g^1/\perp$. *Given a re-encryption key $rk_{x\to g}^{w'}$ and a second-level ciphertext ct_f^2, this algorithm outputs a first-level ciphertext ct_g^1 associated with policy g if $f(x) = 1$ and $w' \models w$, else outputs an error symbol $\perp$.*

Definition 12 (CAB-PRE: Correctness). *A CAB-PRE scheme is correct if:*

[3] In this paper, $f(x) = 1$ implies that the attribute x is accepted by the access policy f, while $f(x) = 0$ indicates that it is not accepted.

– *For all $x \in \{0,1\}^l$ and $f \in \mathcal{F}$ for which $f(x)=1$, and for any condition w and all $\mu \in \{0,1\}$, it holds that*

$$\Pr[\mathsf{Dec}(sk_x, ct_f \in \{ct_f^1, ct_f^2\}) \neq \mu] = \mathsf{negl}(\lambda),$$

where $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$, $ct_f^1 \leftarrow \mathsf{Enc}_1(f,\mu)$ and $ct_f^2 \leftarrow \mathsf{Enc}_2(f,w,\mu)$.

– *For any w' and $rk_{x\to g}^{w'} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$, it holds that*

$$\Pr[\mathsf{Dec}(sk_y, ct_g^1) \neq \mu] = \mathsf{negl}(\lambda),$$

if $g(y)=1$ and $w' \models w$ for $y \in \{0,1\}^l$ and $g \in \mathcal{F}$, where $ct_g^1 \leftarrow \mathsf{ReEnc}(rk_{x\to g}^{w'}, ct_f^2)$.

3.2 Security Model

In the CPA security model for PRE, an adversary can call the oracles of the re-encryption key and re-encryption between pairs of users in honest or in corrupt, or from corrupt to honest, but not from honest to corrupt. The purpose of the HRA security model is that even if the adversary obtains the correctly generated re-encrypted ciphertext, it cannot obtain information about the delegator's secret key. Thus, the adversary in the HRA security definition can query the re-encryption oracle from the honest user to the corrupt user, with the restriction that the original ciphertext must be honestly generated.

The security model of our CAB-PRE scheme primarily concerns HRA. The difficulty of defining the HRA security model for our cryptosystem is how to describe honest and corrupt users and the case where the re-encryption key and the re-encryption oracles terminate, because honest and corrupt users in CPA are defined before the adversary queries the oracle, this setting can be seen as static. However, CAB-PRE uses fine-grained access control, which cannot be set in this way. Moreover, we also would like to achieve adaptive security. How to solve this problem depends on the relation between the attribute and the access policy and whether the adversary could ask for a secret key associated with an attribute. We, therefore, focus on the adversary's queries about the oracles of the re-encryption key and re-encryption. For a re-encryption key query, consider the following relations between the attributes x and y with access policies f and g:

Table 2. Relation between the attributes with access policies in HRA security model

Relation	$x \in \mathsf{K} \wedge y \in \mathsf{K}$	$x \in \mathsf{K} \wedge y \notin \mathsf{K}$	$x \notin \mathsf{K} \wedge y \in \mathsf{K}$	$x \notin \mathsf{K} \wedge y \notin \mathsf{K}$
$f(x)=0 \wedge g(y)=0$	H → H	H → H	H → H	H → H
$f(x)=0 \wedge g(y)=1$	$\boxed{\text{H} \to \text{C}}$	H → H	$\boxed{\text{H} \to \text{C}}$	H → H
$f(x)=1 \wedge g(y)=0$	C → H	C → H	H → H	H → H
$f(x)=1 \wedge g(y)=1$	C → C	C → H	$\boxed{\text{H} \to \text{C}}$	H → H

H → H denotes the pairs of users in honest;
H → C denotes the pairs of users from honest to corrupt;
C → H denotes the pairs of users from corrupt to honest;
C → C denotes the pairs of users in corrupt.

In Table 2, K is a key list to record the attributes and secret keys the adversary has queried; then $x \in \mathsf{K}$ means x has been queried in key generation oracle, and $x \notin \mathsf{K}$ denotes have not. Let's take an example of what this table means. If the the relation $f(x)=0 \wedge g(y)=0$ and $x \in \mathsf{K} \wedge y \in \mathsf{K}$ holds, the pairs of users are honest (*i.e.*, H → H) since attributes x and y are not satisfy policies f and g, respectively. In other words, even if the adversary queries the secret keys of attributes x and y, it has no decryption rights to the ciphertexts associated with f or g. Observed that the adversary cannot query the re-encryption key from the honest user to the corrupt user (*i.e.*, H → C, boxed part of the table) in three cases, namely

$$(f(x)=0 \wedge g(y)=1) \wedge (x \in \mathsf{K} \wedge y \in \mathsf{K}),$$

$$(f(x) = 0 \wedge g(y) = 1) \wedge (x \notin \mathsf{K} \wedge y \in \mathsf{K}),$$
$$(f(x) = 1 \wedge g(y) = 1) \wedge (x \notin \mathsf{K} \wedge y \in \mathsf{K}).$$

Similarly, the re-encryption oracle terminates under the following cases: 1) the conditions in the re-encryption key and the ciphertext do not match; 2) the ciphertext does not contain the condition; 3) $f(x) = 0$, *i.e.*, anyone has a secret key associated with x cannot decrypt the original ciphertext encrypted under f correctly, and 4) the ciphertext submitted by the adversary is a challenge ciphertext, while $g(x) = 1$. Note that excluding the termination condition, the adversary can call the re-encryption oracle. When $f(x) = 1$, the attributes x and y have relations to the access policies f and g, the same as the last two rows of Table 2. Especially under the case of $f(x) = 1 \wedge g(y) = 1$ and $x \notin \mathsf{K} \wedge y \in \mathsf{K}$, the re-encryption oracle must also make a response to the adversary.

Furthermore, Cohen demonstrated in [9] that the HRA-secure PRE scheme also guarantees CPA security. We have the same result for our CAB-PRE scheme, as evidenced in Remark 1.

Now we formally give the HRA security model for our CAB-PRE as follows.

Definition 13 (CAB-PRE: Security Game for HRA). *The adaptive HRA security game of a unidirectional, single-hop CAB-PRE scheme between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. The game consists of three phases as follows.*

Phase 1 (Setup): *This is the setup phase. The challenger generates (pp, msk) by running $\mathsf{Setup}(1^\lambda)$ algorithm and gives the public parameters pp to $\mathcal{A}$. Then, the challenger initializes a counter $\mathsf{numCt} := 0$, a policy-value store $\mathsf{C} := \emptyset$, a key list $\mathsf{K} := \emptyset$ and a set $\mathsf{Deriv} := \emptyset$.*

Phase 2 (Oracle Query): *This is the oracle query phase.*

– $\mathcal{O}_{\mathsf{KeyGen}}(x)$: *For a key query x, the challenger generates $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$ and adds (x, sk_x) in K. It gives sk_x to $\mathcal{A}$.*

– $\mathcal{O}_{\mathsf{Enc}}(f, w, \mu)$: *For an encryption query (f, w, μ), the challenger computes*

$$ct_f \leftarrow \begin{cases} \mathsf{Enc}_1(f, \mu), & \text{if } w = \mathsf{Null} \\ \mathsf{Enc}_2(f, w, \mu), & \text{otherwise} \end{cases}$$

sets $\mathsf{numCt} := \mathsf{numCt}+1$, adds ct_f in C with policy tuple (f, w, numCt), and gives (numCt, ct_f) to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{ReKey}}(x, g, w')$: *For a re-encryption key query (x, g, w'), we assume that there exists an attribute string y. If $y \notin \mathsf{K}$, the challenger records y such that it cannot be queried in $\mathcal{O}_{\mathsf{KeyGen}}$. There exist two cases:*

1) $g(y) = 0$, the challenger generates $rk^{w'}_{x \to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$.

2) $g(y) = 1$, return $\perp$ if for any queried f in the store C such that i) $f(x) = 0$ and $x \in \mathsf{K} \wedge y \in \mathsf{K}$, or ii) $f(x) = 0$ and $x \notin \mathsf{K} \wedge y \in \mathsf{K}$, or iii) $f(x) = 1$ and $x \notin \mathsf{K} \wedge y \in \mathsf{K}$. Otherwise, the challenger produces $rk^{w'}_{x \to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$.

After that, $\mathcal{C}$ gives $rk^{w'}_{x \to g}$ to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{Cha}}(f^*, w^*, (\mu_0, \mu_1))$: *This oracle can only be invoked once. For a challenge query $(f^*, w^*, (\mu_0, \mu_1))$, it requires $f^*(x) = 0$, where $x \in \mathsf{K}$. The challenger flips a bit $b \in \{0, 1\}$, generates*

$$ct_{f^*} \leftarrow \begin{cases} \mathsf{Enc}_1(f^*, \mu_b), & \text{if } w^* = \mathsf{Null} \\ \mathsf{Enc}_2(f^*, w^*, \mu_b), & \text{otherwise} \end{cases}$$

sets $\mathsf{numCt} := \mathsf{numCt} + 1$ and $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. It adds ct_{f^*} in C with policy tuple $(f^*, w^*, \mathsf{numCt})$ and gives $(\mathsf{numCt}, ct_{f^*})$ to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{ReEnc}}((x, g, w'), (f, w, k))$: *For a re-encryption query $((x, g, w'), (f, w, k))$, where $k \leq \mathsf{numCt}$. Assume that there exist an attribute string y. If $y \notin \mathsf{K}$, the challenger records y such that it cannot be queried in $\mathcal{O}_{\mathsf{KeyGen}}$. Then the challenger does the following operations.*

1) If $w' \not\models w$, return $\perp$.

2) If there is no value in C with policy tuple $(f, w \neq \mathsf{Null}, k)$, return $\perp$.

3) If $f(x) = 0$, return $\perp$.

4) If $g(y) = 1 \wedge y \in \mathsf{K} \wedge k \in \mathsf{Deriv}$, return $\perp$.

5) Otherwise, let ct_f be that value in the store C. The challenger produces $ct_g \leftarrow \mathsf{ReEnc}(rk^{w'}_{x \to g}, ct_f)$, where $rk^{w'}_{x \to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$, sets $\mathsf{numCt} := \mathsf{numCt} + 1$, adds ct_g in C with policy tuple $(g, \mathsf{Null}, \mathsf{numCt})$. If $k \in \mathsf{Deriv}$, set $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. Finally, it gives (numCt, ct_g) to $\mathcal{A}$.

Phase 3 (Decision): *This is the decision phase. $\mathcal{A}$ outputs a bit b' for b.*

The advantage of $\mathcal{A}$ winning the adaptive HRA security game is defined as

$$\mathsf{Adv}^{\mathsf{HRA}}_{\mathcal{A},\Pi_{\mathsf{CAB-PRE}}} = |\Pr[b' = b] - 1/2|.$$

Definition 14 (CAB-PRE: Adaptive HRA Security). *Given a security parameter λ, we say the scheme $\Pi_{\mathsf{CAB-PRE}}$ for ciphertext policy is unidirectional, single-hop adaptively HRA-secure if for all PPT adversaries $\mathcal{A}$, there is a negligible function $\mathsf{negl}(\lambda)$ such that*

$$\mathsf{Adv}^{\mathsf{HRA}}_{\mathcal{A},\Pi_{\mathsf{CAB-PRE}}} \leq \mathsf{negl}(\lambda).$$

Remark 1. The HRA security implies CPA security that pointed in [9] for PRE schemes. We have the same conclusion for our CAB-PRE schemes. Based on the above security model, the CPA security model can be captured by performing several modifications that (1) remove the encryption oracle $\mathcal{O}_{\mathsf{Enc}}$, namely, the adversary cannot make any encryption queries and (2) return $\perp$ if $f(x) = 1 \wedge g(y) = 1$ and $x \notin \mathsf{K} \wedge y \in \mathsf{K}$, in $\mathcal{O}_{\mathsf{ReEnc}}$.

4 Adaptively HRA-Secure AB-PRE from LWE

Our method integrates the (ciphertext-policy) ABE scheme [31] with the key switching technique [6]. The resulting scheme is an adaptively HRA-secure (ciphertext-policy) AB-PRE scheme, which resolves the open problem introduced by Susilo et al. [30] in ESORICS'21 about constructing an HRA-secure (ciphertext-policy) AB-PRE scheme. Appendix A provides AB-PRE's syntax and HRA security definition for clarity and completeness. Note that for a ciphertext encrypted under an access policy f, the ciphertext can be converted to the one encrypted under a policy g, where f is arbitrary, as long as the attributes in the re-encryption key satisfy that policy f.

4.1 Technique Overview

The scheme [31] utilizes a conforming constrained pseudorandom function (cPRF) for the predicate class t-CNF (denoted by $\mathcal{F}$ in this paper) and employs circuits $U_{\sigma \to x}$, $U_{\sigma \to f}$, and $U_{f \to x}$. The public parameters of the scheme consist of two matrices $\mathbf{A}$ and $\mathbf{B}$, a vector $\mathbf{v}$, and the cPRF's public parameters $\mathsf{P.pp}$. The master secret key comprises a trapdoor of matrix $\mathbf{B}$ and the cPRF's master secret key σ. For any policy $f \in \mathcal{F}$, one obtains an efficient computable matrix $\mathbf{A}_f$ using lattice evaluation. The ciphertext is a Dual-Regev encryption with respect to the tuple $(\mathbf{B} \| [\mathbf{A}_f - s_f \otimes \mathbf{G}], \mathbf{v})$, where $s_f \leftarrow \mathsf{P.KeySim}(P.pp, f)$. Similarly, one can capture an efficient computable matrix $\mathbf{A}_{x,r}$ using lattice evaluation for each attribute string x. The secret key is a short vector $\mathbf{k}$ sampled from the $\mathsf{SamplePre}$ algorithm and satisfies $(\mathbf{B} \| \mathbf{A}_{x,r})\mathbf{k} = \mathbf{v}$, where $r \leftarrow \mathsf{P.Eval}(\sigma, x)$. The decryption algorithm computes $r' \leftarrow U_{f \to x}(s_f)$ and successfully recovers the message by the secret key $\mathbf{k}$ if and only if (*i.e.*, iff) $f(x) = 1 \wedge r \neq r'$.

We employ a key switching technique that generates a re-encryption key

$$\mathbf{Z} = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{v} + \mathbf{r}_3 - \mathsf{P2}(\mathbf{k}) \\ \mathbf{0}_{1 \times (m' + m l_g)} & 1 \end{pmatrix}$$

to construct an AB-PRE scheme. The encryption tuple $(\mathbf{H}, \mathbf{v}) = (\mathbf{B} \| [\mathbf{A}_g - s_g \otimes \mathbf{G}], \mathbf{v})$ relates to the policy $g \in \mathcal{F}$, and $\mathbf{R}_1, \mathbf{R}_2, \mathbf{r}_3$ denote error terms. It is noteworthy that secret keys produce re-encryption keys. However, in the HRA security proof, the challenger must respond to re-encryption queries made by the adversary, even in the absence of secret keys. Choose a small value $\delta \xleftarrow{\$} \chi$, and a short vector $\mathbf{d}$ is sampled such that $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{d} = \mathbf{v}\cdot\delta$. This results in creating a modified re-encryption key, denoted as

$$\mathbf{Z}' = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{v} + \mathbf{r}_3 - \mathsf{P2}(\mathbf{d}) \\ \mathbf{0}_{1\times(m'+ml_g)} & \delta \end{pmatrix}.$$

By using this modified re-encryption key, a re-encrypted ciphertext can be obtained. However, it is essential to note that this approach impacts the noise bound of the transformed ciphertext during decryption.

To prove the adaptive HRA security of the AB-PRE scheme, we introduce two randomized algorithms, KeyGen_1 and KeyGen_2, which play a crucial role in handling re-encryption queries. These algorithms enable us to simulate re-encryption keys without requiring access to secret keys. However, if the adversary has decryption capabilities, the master secret key must be utilized. Specifically, we employ the $\mathsf{SamplePre}$ algorithm to generate a small vector $\mathbf{k}$ (*s.t.*, $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{k} = \mathbf{v}$) and produce an extended trapdoor in the key generation algorithm. We subsequently sample a short vector $\mathbf{d}$ (*s.t.*, $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{d} = \mathbf{v}\cdot\delta$) based on the extended trapdoor to generate a re-encryption key. Since $\mathbf{d}$ is not a derivative of $\mathbf{k}$ [4], it is possible to simulate a re-encryption key directly derived from the master secret key without relying on the secret key. One can satisfy the above requirement using algorithms with "symmetric" properties, such as $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$. To ensure the security of our scheme, we reduce it to the hardness of the LWE problem. Specifically, we embed an LWE instance $(\mathbf{B}\|\mathbf{v}, \mathbf{u}_0\|u_2')$ into the challenge ciphertext $(\mathbf{u}_0, u_2 = u_2' + \mu_b\lfloor q/2\rceil)$, making it indistinguishable from the uniform distribution for any potential adversary.

4.2 Construction

Let $\mathsf{P} = (\mathsf{P.Setup}, \mathsf{P.Eval}, \mathsf{P.Constrain}, \mathsf{P.ConstrainEval})$ be a conforming cPRF for a class family $\mathcal{F}$ of t-CNF predicates with input length l and output length k. Assume that the master secret key length of P is λ. For all $f \in \mathcal{F}$, let l_f denote the size of constrained key which can be computed efficiently given the function f and $\mathsf{P.Constrain}$ algorithm. Note that $U_{\sigma\to x}, U_{\sigma\to f}$ and $U_{f\to x}$ are circuits defined as Definition 2. Define an adaptively HRA-secure (ciphertext-policy) AB-PRE as follows.

$\mathsf{Setup}(1^\lambda)$: Run $(P.pp, P.msk) \leftarrow \mathsf{P.Setup}(1^\lambda)$, set $\sigma = P.msk$. $n, q, m', \tau, \chi, \tilde{\chi}$ are parameters and let $m = n\lceil \log q\rceil$. Invoke $(\mathbf{B}, \mathbf{T}_\mathbf{B}) \leftarrow \mathsf{TrapGen}(1^n, m', q)$. Sample a matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m\lambda}$ and a vector $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$. Output $pp = (\mathbf{B}, \mathbf{A}, \mathbf{v}, P.pp)$ and $msk = (\sigma, \mathbf{T}_\mathbf{B})$.

$\mathsf{KeyGen}(msk, x)$: Compute $\mathbf{H}_{\sigma\to x} \leftarrow \mathsf{EvalF}(U_{\sigma\to x}, \mathbf{A})$, let $\mathbf{A}_x = \mathbf{A}\mathbf{H}_{\sigma\to x}$. Evaluate $r \leftarrow \mathsf{P.Eval}(\sigma, x)$ and $\mathbf{H}_r \leftarrow \mathsf{EvalF}(I_r, \mathbf{A}_x)$, where $I_r : \{0,1\}^k \to \{0,1\}$ is a function that on input r' returns 0 iff $r \neq r'$. Set $\mathbf{A}_{x,r} = \mathbf{A}_x\mathbf{H}_r$. Capture $\mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}} \leftarrow \mathsf{RandBasis}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathsf{ExtendRight}(\mathbf{B}, \mathbf{T}_\mathbf{B}, \mathbf{A}_{x,r}), \tau)$ and sample a short $\mathbf{k} \leftarrow \mathsf{SamplePre}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{v}, \tau)$ such that $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{k} = \mathbf{v}$. Output $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{k})$.

$\mathsf{Enc}(f, \mu)$: Compute $s_f \leftarrow \mathsf{P.KeySim}(P.pp, f)$. Choose randomly $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}_0 \xleftarrow{\$} \chi^{m'}$, $\mathbf{e}_1 \xleftarrow{\$} \tilde{\chi}^{ml_f}$, $e_2 \xleftarrow{\$} \chi$, set

$$\mathbf{u}_0 = \mathbf{s}^T\mathbf{B} + \mathbf{e}_0^T,$$

$$\mathbf{u}_1 = \mathbf{s}^T[\mathbf{A}_f - s_f \otimes \mathbf{G}] + \mathbf{e}_1^T,$$

$$u_2 = \mathbf{s}^T\mathbf{v} + e_2 + \mu\lfloor q/2\rceil,$$

where $\mathbf{A}_f = \mathbf{A}\mathbf{H}_{\sigma\to f}$ and $\mathbf{H}_{\sigma\to f} \leftarrow \mathsf{EvalF}(U_{\sigma\to f}, \mathbf{A})$. Output $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, u_2)$.

[4] Our proposed scheme relies on a crucial requirement that the norms of the vectors $\mathbf{k}$ and $\mathbf{d}$ should be bounded by $\tau\sqrt{m'+m}$ since they are generated from the Gaussian sampling with parameter τ. If $\mathbf{d}$ is a derivative of $\mathbf{k}$, *i.e.*, $\mathbf{d} = \mathbf{k}\cdot\delta$, then we can bound its infinity norm as $\|\mathbf{d}\|_\infty \le \tau\sqrt{m'+m}\cdot B$. This inequality $\tau\sqrt{m'+m} < \|\mathbf{d}\|_\infty \le \tau\sqrt{m'+m}\cdot B$ holds with a non-negligible probability, where $\tau \in O(\lambda, n, k, (2m)^{d+3})$ and $B = B(n)$.

$\mathsf{Dec}(sk_x, ct_f)$**:** Parse $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{k})$ and $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, u_2)$. Compute $r' \leftarrow U_{f\to x}(s_f)$, then abort if $r' = r$. Otherwise, capture $\mathbf{A}_x$ and $\mathbf{A}_f$ are the same as in KeyGen and Enc, respectively. Evaluate

$$\hat{\mathbf{H}}_{r,r'} \leftarrow \mathsf{EvalFX}(I_r, r', \mathbf{A}_x),$$

$$\hat{\mathbf{H}}_{s_f\to r'} \leftarrow \mathsf{EvalFX}(U_{f\to x}, s_f, \mathbf{A}_f).$$

Lastly, compute

$$u = u_2 - (\mathbf{u}_0 \| \mathbf{u}_1 \hat{\mathbf{H}}_{s_f\to r'} \hat{\mathbf{H}}_{r,r'})\mathbf{k}.$$

Output 1 iff $|u| > q/4$, otherwise output 0. It is worth noting that the decryption of a transformed ciphertext involves a coefficient factor δ, and the correct recovery of the message μ is contingent upon the appropriate tailoring of the noise bound.

$\mathsf{ReKeyGen}(sk_x, g)$**:** Parse $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{k})$. Select a function $g \in \mathcal{F}$ that conforms to gradual evaluation (as specified in Definition 2) and where the size of the constrained key is l_g. Sample $s_g \leftarrow \mathsf{P.KeySim}(P.pp, g)$. Let $\mathbf{H} = \mathbf{B}\|[\mathbf{A}_g - s_g \otimes \mathbf{G}]$, where $\mathbf{A}_g = \mathbf{A}\mathbf{H}_{\sigma\to g}$ and $\mathbf{H}_{\sigma\to g} \leftarrow \mathsf{EvalF}(U_{\sigma\to g}, \mathbf{A})$. Compute $\mathbf{A}_{x,r}$ the same as in KeyGen. Set $\mathbf{v}' = \mathbf{v}\cdot\delta$, where $\delta \xleftarrow{\$} \chi$, sample a short $\mathbf{d}$ by running $\mathsf{SamplePre}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{v}', \tau)$ algorithm such that $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{d} = \mathbf{v}'$. Select randomly $\mathbf{R}_1 \in \chi^{(m'+m)\lceil\log q\rceil\times n}, \mathbf{R}_2 \in \chi^{(m'+m)\lceil\log q\rceil\times(m'+ml_g)}$ and $\mathbf{r}_3 \in \chi^{(m'+m)\lceil\log q\rceil}$, compute

$$\mathbf{Z} = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{v} + \mathbf{r}_3 - \mathsf{P2}(\mathbf{d}) \\ \mathbf{0}_{1\times(m'+ml_g)} & \delta \end{pmatrix}.$$

Output $rk_{x\to g} = (s_g, r, \delta, \mathbf{Z})$.

$\mathsf{ReEnc}(rk_{x\to g}, ct_f)$**:** Parse $rk_{x\to g} = (s_g, r, \delta, \mathbf{Z})$ and $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, u_2)$. Compute $r' \leftarrow U_{f\to x}(s_f)$, then abort if $r' = r$. Otherwise, capture $\mathbf{A}_x$, $\mathbf{A}_f$, $\hat{\mathbf{H}}_{r,r'}$ and $\hat{\mathbf{H}}_{s_f\to r'}$ the same as in $\mathsf{KeyGen}, \mathsf{Enc}$ and Dec, respectively. Finally, let $\mathbf{u}_1' = \mathbf{u}_1\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'}$, evaluate

$$ct_{f\to g} = (\mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\|u_2)\cdot\mathbf{Z}.$$

Output $ct_g = (s_g, \delta, ct_{f\to g})$.

4.3 Correctness and Choice of Parameters

Theorem 2. *The AB-PRE scheme is correct with respect to f under proper parameters as below.*

Proof. First, we direct our focus towards the correct decryption of the original ciphertext. For the security parameter λ, consider functions $f \in \mathcal{F}$ and I_r, two strings $x \in \{0,1\}^l$ and $r' \in \{0,1\}^k$ such that $f(x) = 1 \wedge I_r(r') = 0$, if $(pp, msk) \leftarrow \mathsf{Setup}(1^\lambda)$, $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$ and $ct_f \leftarrow \mathsf{Enc}(f, \mu)$, then we have $\mu \leftarrow \mathsf{Dec}(sk_x, ct_f)$. As demonstrated by [31], the probability of $r = r'$ is almost negligible, as evidenced by a reduction to the pseudorandomness game of the conforming cPRF P. Specifically, we have $r \leftarrow \mathsf{P.Eval}(P.msk, x)$ and $r' \leftarrow U_{f\to x}(s_f)$ (*i.e.*, r' is computed as $\mathsf{P.ConstrainEval}(s_f, x)$, where $s_f \leftarrow \mathsf{P.KeySim}(P.pp, f)$). Given the function I_r, the existence of $I_r(r') = 0$ can be established with overwhelming advantage.

When $f(x) = 1$, the property of gradual evaluation of P (see Definition 2) shows that the effective sub-circuit of $U_{f\to x} \circ U_{\sigma\to f}$ is equivalent to the circuit $U_{\sigma\to x}$. By Theorem 1, it follows that $\mathbf{H}_{\sigma\to x} = \mathbf{H}_{\sigma\to f}\mathbf{H}_{f\to x}$. Hence we have $\mathbf{A}_x = \mathbf{A}\mathbf{H}_{\sigma\to x} = \mathbf{A}\mathbf{H}_{\sigma\to f}\mathbf{H}_{f\to x} = \mathbf{A}_f\mathbf{H}_{f\to x}$, where $\mathbf{H}_{f\to x} \leftarrow \mathsf{EvalF}(U_{f\to x}, \mathbf{A})$ and $\mathbf{H}_{\sigma\to f} \leftarrow \mathsf{EvalF}(U_{\sigma\to f}, \mathbf{A})$.

Based on Theorem 1, we capture $(\mathbf{H}_r, \hat{\mathbf{H}}_{r,r'})$ and $(\mathbf{H}_{f\to x}, \hat{\mathbf{H}}_{s_f\to r'})$. Precisely,

$$\mathbf{H}_r \leftarrow \mathsf{EvalF}(I_r, \mathbf{A}_x),$$

$$\hat{\mathbf{H}}_{r,r'} \leftarrow \mathsf{EvalFX}(I_r, r', \mathbf{A}_x),$$

$$\hat{\mathbf{H}}_{s_f\to r'} \leftarrow \mathsf{EvalFX}(U_{f\to x}, s_f, \mathbf{A}_f).$$

Then, we compute

$$\begin{aligned}[\mathbf{A}_f - s_f \otimes \mathbf{G}]\hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'} &= [\mathbf{A}_f \mathbf{H}_{f\to x} - U_{f\to x}(s_f) \otimes \mathbf{G}]\hat{\mathbf{H}}_{r,r'} \\ &= [\mathbf{A}_x - r' \otimes \mathbf{G}]\hat{\mathbf{H}}_{r,r'} \\ &= \mathbf{A}_x \mathbf{H}_r - I_r(r') \otimes \mathbf{G} \\ &= \mathbf{A}_{x,r}.\end{aligned}$$

Thus,

$$\begin{aligned}u_2 - (\mathbf{u}_0 \| \mathbf{u}_1 \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'})\mathbf{k} &= u_2 - (\mathbf{u}_0 \| \mathbf{s}^T \mathbf{A}_{x,r} + \mathbf{e}_1^T \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'})\mathbf{k} \\ &= u_2 - \mathbf{s}^T(\mathbf{B} \| \mathbf{A}_{x,r})\mathbf{k} - (\mathbf{e}_0^T \| \mathbf{e}_1^T \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'})\mathbf{k} \\ &= \mu \lfloor q/2 \rceil + e_2 - (\mathbf{e}_0^T \| \mathbf{e}_1')\mathbf{k},\end{aligned}$$

where $\mathbf{e}_1' = \mathbf{e}_1^T \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'}$.

The depths of $U_{\sigma \to f}$ and $U_{f \to x}$ are denoted by d_{Con} and d_{ConEval}, respectively. These depths are limited by the depth $d = \mathsf{poly}(\lambda)$ of $U_{\sigma \to x}$, as P serves as the gradual depth of $U_{\sigma \to f}$ and $U_{f \to x}$. Note that

$$\begin{aligned}\|\mathbf{e}_1'\|_\infty &\le m^2 l_f k \|\mathbf{e}_1^T\|_\infty \|\hat{\mathbf{H}}_{s_f \to r'}\|_\infty \|\hat{\mathbf{H}}_{r,r'}\|_\infty \\ &\le m^2 l_f k \tilde{B}(2m)^{d_{\mathsf{ConEval}}+1}\end{aligned}$$

and

$$\|\mathbf{k}\|_\infty \le \tau\sqrt{m'+m},$$

due to $\mathbf{e}_1 \in \tilde{\chi}^{ml_f}, \hat{\mathbf{H}}_{s_f \to r'} \in \mathbb{Z}^{ml_f \times mk}, \hat{\mathbf{H}}_{r,r'} \in \mathbb{Z}^{mk \times m}$ and the tailcut inequality (see Definition 4) of the discrete Gaussian.

Therefore, if $m', l_f, k \in O(n, \lceil \log q \rceil)$, $\tilde{B} \in O(B, n)$ and $\tau \in O(\lambda, k, (2m)^{d+3})$, then

$$\begin{aligned}|e_2 - (\mathbf{e}_0^T \| \mathbf{e}_1')\mathbf{k}| &\le |e_2| + (m'\|\mathbf{e}_0^T\|_\infty + m\|\mathbf{e}_1'\|_\infty)\|\mathbf{k}\|_\infty \\ &\le B + (m'B + m^3 l_f k \tilde{B}(2m)^{d_{\mathsf{ConEval}}+1})\tau\sqrt{m'+m} \\ &\le B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}.\end{aligned}$$

To capture correct decryption, the magnitude should be less than $q/4$, *i.e.*, $|e_2 - (\mathbf{e}_0^T \| \mathbf{e}_1')\mathbf{k}| < q/4$. Choosing the parameters

- q, χ, B as Corollary 1 and note that $q \le 2^n$ and $q/B \ge 2^{n^\epsilon}$.
- $n \ge \lambda$ such that $(2n^2)^{(2d+4)} \le 2^{n^\epsilon}$, where $\epsilon \in (0,1)$ and $n \le d^{O(1/\epsilon)}$.
- $m' = (n+1)\lceil \log q \rceil + 2\lambda$, $B' = m' m \lambda B (2m)^{d_{\mathsf{Con}}}$, $E' \le 2^{n^\epsilon}$.

Let

$$E = B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}$$

and

$$E' = 4E/B = 4 \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}.$$

Since $E' < q/B$, then $E = BE'/4 < q/4$. Therefore,

$$|e_2 - (\mathbf{e}_0^T \| \mathbf{e}_1')\mathbf{k}| \le B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4} < q/4$$

is overwhelming, and the decryption of the original ciphertext is correct.

Next, we show the correctness of the re-encrypted ciphertext. Given an original ciphertext associated with function f, it can be efficiently re-encrypted to another one associated with function g using a re-encryption key. Specifically, parse the original ciphertext $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, u_2)$ and the re-encryption key $rk_{x \to g} = (s_g, r, \sigma, \mathbf{Z})$, the re-encryption process in the following way:

If $f(x) = 1 \wedge I_r(r') = 0$ holds, where $r' \leftarrow U_{f \to x}(s_f)$, we compute

$$\mathbf{u}_1' = \mathbf{u}_1 \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'} = \mathbf{s}^T \mathbf{A}_{x,r} + \mathbf{e}_1^T \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'},$$

then

$$\begin{aligned} ct_{f\to g} &= (\mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\|u_2)\cdot \mathbf{Z} \\ &= (\mathbf{c}_0\|\mathbf{c}_1 \quad c_2). \end{aligned}$$

Let $\bar{\mathbf{s}} = (\mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\mathbf{R}_1)^T$, we have

$$\begin{aligned} \mathbf{c}_0\|\mathbf{c}_1 &= \bar{\mathbf{s}}^T(\mathbf{B}\|[\mathbf{A}_g - s_g\otimes \mathbf{G}]) + \mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\mathbf{R}_2 \\ &= \bar{\mathbf{s}}^T(\mathbf{B}\|[\mathbf{A}_g - s_g\otimes \mathbf{G}]) + (\bar{\mathbf{e}}_0^T\|\bar{\mathbf{e}}_1^T) \end{aligned}$$

and

$$c_2 = \bar{\mathbf{s}}^T\mathbf{v} + \bar{e}_2 + \mu\lfloor q/2\rceil\cdot\delta,$$

where $\bar{e}_2 = \mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\mathbf{r}_3 - (\mathbf{e}_0^T\|\mathbf{e}_1')\mathbf{d} + e_2\cdot\delta$ and $|\bar{e}_2| \le B\cdot \mathsf{poly}(n, \lceil\log q\rceil)\cdot(2m)^{d_{\mathsf{ConEval}}+d+4}$.

Without loss of generality, let $sk_y = (\bar{r}, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{y,\bar{r}}}, \bar{\mathbf{k}})$ denote a secret key for an attribute string y that satisfies $g(y) = 1$ and $I_{\bar{r}}(\bar{r}') = 0$, where $\bar{r}' \leftarrow U_{g\to y}(s_g)$. To capture correct decryption, we compute

$$u = c_2 - (\mathbf{c}_0\|\mathbf{c}_1\hat{\mathbf{H}}_{s_g\to\bar{r}'}\hat{\mathbf{H}}_{\bar{r},\bar{r}'})\bar{\mathbf{k}}.$$

Based on our parameter settings and analysis, the norm of error term is bounded by

$$|\bar{e}_2^T - (\bar{\mathbf{e}}_0^T\|\bar{\mathbf{e}}_1')\bar{\mathbf{k}}| \le 2B\cdot\mathsf{poly}(n\lceil\log q\rceil)\cdot(2m)^{d_{\mathsf{ConEval}}+d+4} < q/4\cdot\delta$$

in which $\bar{\mathbf{e}}_1' = \bar{\mathbf{e}}_1^T\hat{\mathbf{H}}_{s_g\to\bar{r}'}\hat{\mathbf{H}}_{\bar{r},\bar{r}'}$ with an overwhelming probability, and the decryption of the re-encrypted ciphertext is correct. □

4.4 Security Proof

For now, we will construct two efficient randomized algorithms KeyGen_1 and KeyGen_2 that are the heart of the HRA security proof.

- $\mathsf{KeyGen}_1(msk, x) \to \hat{sk}_x$: Taking as input $msk = (\sigma, \mathbf{T}_\mathbf{B})$ and $x \in \{0,1\}^l$, the algorithm computes $\mathbf{A}_{x,r}$ the same as in KeyGen, and captures $\hat{sk}_x$ by calling $\mathsf{SampleLeft}(\mathbf{B}, \mathbf{A}_{x,r}, \mathbf{T}_\mathbf{B}, \mathbf{v}', \tau)$, for some terms $\mathbf{B}$ and $\mathbf{v}'$.

- $\mathsf{KeyGen}_2(msk, x) \to \hat{sk}_x$: Taking as input $msk = \sigma$ (without $\mathbf{T}_\mathbf{B}$) and $x \in \{0,1\}^l$, the algorithm computes $\mathbf{A}_{x,r}$ the same as in KeyGen, and captures $\hat{sk}_x$ by calling $\mathsf{SampleRight}(\mathbf{B}, \mathbf{G}, \mathbf{R}, \mathbf{T}_\mathbf{G}, \mathbf{v}', \tau)$ if $\mathbf{A}_{x,r} = \mathbf{BR} + \mathbf{G}$, for some terms $\mathbf{R}$ and $\mathbf{v}'$.

Note that by Lemma 3 and Lemma 4, the outputs of $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ are distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{v}'}(\mathbf{B}\|\mathbf{A}_{x,r}),\tau}$. Thus, sampling $\hat{sk}_x$ from KeyGen_1 or KeyGen_2 are statistically indistinguishable.

Theorem 3. *For a class family $\mathcal{F}$, if P is a conforming cPRF, then AB-PRE is a unidirectional, single-hop, adaptively HRA-secure scheme under the hardness of DLWE$_{n,q,\chi}$ problem.*

Proof. The adaptive HRA security model of AB-PRE is identical to Definition 13 except for canceling some condition requirements (*i.e.*, all conditions are matched by default). Let $\mathcal{A}$ be a PPT adversary that breaks the adaptive HRA security of AB-PRE. We show that the proof proceeds in a sequence of games. In each game, we define S_i to be the event that $\mathcal{A}$ wins in $\mathbf{Game}_i$.

$\mathbf{Game}_0$: This is the original security game, in which most operations are identical to the real scheme for the adversary's queries. Nonetheless, special attention is directed towards the responses to the re-encryption key queries, given the stringent constraints and subtle transformations involved, which may differ from the real scheme but retain an equivalent relation. The handling of these queries is expounded as follows:

– When $\mathcal{A}$ makes a re-encryption key query on (x, g), there exist an attribute string y such that $g(y) = 0$. The challenger chooses a matrix $\mathbf{Z}_1 \xleftarrow{\$} \chi^{(m'+m)\lceil \log q \rceil \times (m'+ml_g)}$ and a vector $\mathbf{z}_2 \xleftarrow{\$} \chi^{(m'+m)\lceil \log q \rceil}$, simulates the fourth entries of the re-encryption key as

$$\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_1 & \mathbf{z}_2 \\ \mathbf{0}_{1\times(m'+ml_g)} & \delta \end{pmatrix},$$

where $\delta \xleftarrow{\$} \chi$. It returns $rk_{x\to g}$ to $\mathcal{A}$.

– When $\mathcal{A}$ makes a re-encryption key query on (x, g), there exist an attribute string y such that $g(y) = 1$. In addition to the termination conditions described in Definition 13, the challenger first invokes $\mathsf{KeyGen}_1(msk, x)$ algorithm (here $msk = (\sigma, \mathbf{T_B})$) to obtain $\hat{sk}_x (= \mathbf{d})$ without generating the secret key sk_x. It then computes $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(\hat{sk}_x, g)$ and gives $rk_{x\to g}$ to $\mathcal{A}$.

By Definition 13 and Definition 14, we have $|\Pr[S_0] - 1/2| \leq \mathsf{negl}(\lambda)$.

Remark 2. Recall in the real scheme, the challenger creates

$$\mathbf{Z} = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{v} + \mathbf{r}_3 - \mathsf{P2}(\mathbf{d}) \\ \mathbf{0}_{1\times(m'+ml_g)} & \delta \end{pmatrix},$$

where $\mathbf{R}_1, \mathbf{R}_2, \mathbf{r}_3$ are sampled from discrete Gaussian distributions. Observe that $\mathbf{Z}$ is indistinguishable with uniform distribution (*i.e.*, satisfying key privacy [3]) based on the hardness of HNF-LWE problem (see Definition 7, more proof details please refer to [21]). Consequently, the challenger could return a random matrix that is identically and independently distributed to the real re-encryption key distribution.

Game$_1$: In this game, we change $r \xleftarrow{\$} \{0,1\}^k$ instead of $r \leftarrow \mathsf{P.Eval}(\sigma, x)$ when generating sk_x, if the challenge ciphertext or its derivatives (*i.e.*, $k \in \mathsf{Deriv}$) are queried in $\mathcal{O}_{\mathsf{ReEnc}}$. Based on the pseudorandomness game of P, we have $|\Pr[S_0] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$.

Game$_2$: In this game, we change the way s_{f^*} for a target function f^* is created. Concretely, instead of computing $s_{f^*} \leftarrow \mathsf{P.KeySim}(P.pp, f^*)$, the challenger generates $s_{f^*} \leftarrow \mathsf{P.Constrain}(\sigma, f^*)$. We show $|\Pr[S_1] - \Pr[S_2]| = \epsilon_{\mathsf{KeySim}}$, where $\epsilon_{\mathsf{KeySim}}$ is the advantage of breaking key simulation game of P and this is negligible. Suppose there exist an adversary $\mathcal{A}_0$ such that $|\Pr[S_1] - \Pr[S_2]|$ is non-negligible, we can build an algorithm $\mathcal{C}$ that wins the key simulation game of P with an overwhelming advantage.

1) At the beginning, $\mathcal{C}$ receives $P.pp$. Then, it generates pp and msk as in the previous game.
2) Upon input a bit string $x \in \{0,1\}^l$ for $\mathcal{O}_{\mathsf{KeyGen}}(x)$, $\mathcal{C}$ captures r_x by sending x to the evaluation oracle of the key simulation game. Let $r = r_x$, it answers sk_x as in the previous game.
3) Receive a challenge tuple $(f^*, (\mu_0, \mu_1))$. $\mathcal{C}$ sends f^* to the challenge oracle of the key simulation game.
4) Receive sk_{f^*} and set $s_{f^*} = sk_{f^*}$. $\mathcal{C}$ computes ct_{f^*} the same as in the previous game and returns it to $\mathcal{A}_0$.
5) Answer the subsequent queries as in Step 2).
6) $\mathcal{A}_0$ guesses it is communicating with a **Game**$_1$ or **Game**$_2$ challenger. At last, $\mathcal{C}$ outputs $\mathcal{A}_0$'s guess as the answer to the key simulation game challenge it is trying to distinguish.

If the challenger chooses $b = 1$ in the key simulation game, $\mathcal{C}$ provides a view of **Game**$_1$ to $\mathcal{A}_0$. Otherwise, $\mathcal{C}$ provides a view of **Game**$_2$ to $\mathcal{A}_0$. In other words, any advantage that $\mathcal{A}_0$ distinguishes between these two games translates to an identical advantage in the key simulation game. Therefore, if $|\Pr[S_1] - \Pr[S_2]|$ is non-negligible, then $\mathcal{C}$ could break the key simulation game with a non-negligible advantage.

Remark 3. The re-encryption of challenge ciphertext or its derivatives can be achieved under certain constraints, specifically $f^*(x) = 1$ and at least one of $g(y) = 0$ and $y \notin \mathsf{K}$ holds, where y is an attribute string. However, a contradiction arises as $f^*(x) = 0$ is required in the challenge oracle, which prohibits the querying of x in $\mathcal{O}_{\mathsf{KeyGen}}$. In $\mathbf{Game}_0$, the challenger implicitly generates the corresponding re-encryption key $rk_{x\to g}$ under $f^*(x) = 1$, just unknown to the adversary. The probability of $r \neq r'$ is non-negligible in $\mathbf{Game}_0$, but the conflict arises as $r = r'$ in a significant probability when $r' \leftarrow \mathsf{P.ConstrainEval}(s_{f^*}, x)$ and $r \leftarrow \mathsf{P.Eval}(\sigma, x)$, where $s_{f^*} \leftarrow \mathsf{P.Constrain}(\sigma, f^*)$ (*i.e.*, the change we made in $\mathbf{Game}_2$). To address this issue and enable the re-encryption of challenge ciphertext or its derivatives in $\mathbf{Game}_2$, we modify the sources of r as demonstrated in $\mathbf{Game}_1$.

$\mathbf{Game}_3$: In this game, we change the way the matrix $\mathbf{A}$ is generated. Recall in the previous game, the challenger chooses $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m\lambda}$. Now it first samples a matrix $\mathbf{R} \xleftarrow{\$} \{1, -1\}^{m'\times m\lambda}$ and sets $\mathbf{A} = \mathbf{BR} + \sigma \otimes \mathbf{G}$. Since $m' \geq (n + 1)\lceil\log q\rceil + 2\lambda$ and the generalized leftover hash lemma (see Definition 10), the distribution $(\mathbf{B}, \mathbf{BR})$ is statistically indistinguishable to the distribution $(\mathbf{B}, \mathbf{U})$, where $\mathbf{U}$ is a random matrix in $\mathbb{Z}_q^{n\times m\lambda}$. Thus, we have $|\Pr[S_2] - \Pr[S_3]| \leq \mathsf{negl}(\lambda)$.

$\mathbf{Game}_4$: In this game, we change again the way challenge query f^* is answered and the way of generating $\mathbf{u}_1^*$. Concretely, when $\mathcal{A}$ makes a challenge query for $(f^*, (\mu_0, \mu_1))$, the challenger computes

$$\begin{aligned}\mathbf{A}_{f^*} - s_{f^*} \otimes \mathbf{G} &= \mathbf{AH}_{\sigma\to f^*} - U_{\sigma\to f^*}(\sigma) \otimes \mathbf{G}\\ &= [\mathbf{A} - \sigma \otimes \mathbf{G}]\hat{\mathbf{H}}_{msk\to s_{f^*}}\\ &= \mathbf{BR}\hat{\mathbf{H}}_{msk\to s_{f^*}},\end{aligned}$$

where $\hat{\mathbf{H}}_{msk\to s_{f^*}} \leftarrow \mathsf{EvalFX}(U_{\sigma\to f^*}, \sigma, \mathbf{A})$. The way it generates $\mathbf{u}_0^*$ and u_2^* remains unaltered. Recall in the previous game, by sampling $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{e}_1 \xleftarrow{\$} \tilde{\chi}^{ml_f}$, the challenger computes $\mathbf{u}_1^* = \mathbf{s}^T[\mathbf{A}_{f^*} - s_{f^*} \otimes \mathbf{G}] + \mathbf{e}_1^T$. Now, $\mathbf{u}_1^*$ will be substituted as

$$\begin{aligned}\mathbf{u}_1^* &= \mathbf{u}_0^*\mathbf{R}\hat{\mathbf{H}}_{msk\to s_{f^*}} + \mathbf{e}_1^T\\ &= (\mathbf{s}^T\mathbf{B} + \mathbf{e}_0^T)\mathbf{R}\hat{\mathbf{H}}_{msk\to s_{f^*}} + \mathbf{e}_1^T\\ &= \mathbf{s}^T[\mathbf{A}_{f^*} - s_{f^*} \otimes \mathbf{G}] + \mathbf{e}_0^T + \mathbf{R}\hat{\mathbf{H}}_{msk\to s_{f^*}} + \mathbf{e}_1^T,\end{aligned}$$

where $\mathbf{e}_0 \xleftarrow{\$} \chi^{m'}$. Note that

$$B' = \|\mathbf{e}_0^T + \mathbf{R}\hat{\mathbf{H}}_{msk\to s_f}\|_\infty \leq m'm\lambda\|\mathbf{e}_0^T\|_\infty\|\mathbf{R}\|_\infty\|\hat{\mathbf{H}}_{msk\to s_{f^*}}\|_\infty \leq m'm\lambda B(2m)^{d_{\mathsf{Con}}},$$

in which d_{Con} denotes the depth of $U_{\sigma\to f^*}$. By Definition 6, let $\tilde{\chi}$ be B'-swallowing, it holds that $\mathbf{u}_1^*$ generated by two methods are within a negligible statistical distance. Therefore, we have $|\Pr[S_3] - \Pr[S_4]| \leq \mathsf{negl}(\lambda)$.

$\mathbf{Game}_5$: In this game, we change the way key queries are answered. When $\mathcal{A}$ queries on x, the challenger evaluates $r \leftarrow \mathsf{P.Eval}(\sigma, x)$ and $\hat{\mathbf{H}}_{msk\to r} \leftarrow \mathsf{EvalFX}(U_{\sigma\to x}, \sigma, \mathbf{A})$, and computes

$$\begin{aligned}[\mathbf{A} - \sigma \otimes \mathbf{G}]\hat{\mathbf{H}}_{msk\to r} &= \mathbf{AH}_{\sigma\to x} - U_{\sigma\to x}(\sigma) \otimes \mathbf{G}\\ &= \mathbf{AH}_{\sigma\to x} - r \otimes \mathbf{G}\\ &= \mathbf{A}_x - r \otimes \mathbf{G}.\end{aligned}$$

Then, we have

$$[\mathbf{A}_x - r \otimes \mathbf{G}]\hat{\mathbf{H}}_{r,r} = \mathbf{A}_x\mathbf{H}_r - I_r(r) \otimes \mathbf{G} = \mathbf{A}_{x,r} - \mathbf{G}$$

since $I_r(r) = 1$ and $\hat{\mathbf{H}}_{r,r} \leftarrow \mathsf{EvalFX}(I_r, r, \mathbf{A}_x)$. Therefore, it holds that $\mathbf{BR}\hat{\mathbf{H}}_{msk\to r}\hat{\mathbf{H}}_{r,r} = \mathbf{A}_{x,r} - \mathbf{G}$ due to $\mathbf{A} - \sigma \otimes \mathbf{G} = \mathbf{BR}$. Note that

$$\mathbf{B}\|\mathbf{A}_{x,r} = \mathbf{B}\|(\mathbf{BR}\hat{\mathbf{H}}_{msk\to r}\hat{\mathbf{H}}_{r,r} + \mathbf{G})$$

and

$$\|\mathbf{R}\hat{\mathbf{H}}_{msk\to r}\hat{\mathbf{H}}_{r,r}\|_\infty \leq m^2\lambda k\|\mathbf{R}\|_\infty\|\hat{\mathbf{H}}_{msk\to r}\|_\infty\|\hat{\mathbf{H}}_{r,r}\|_\infty \leq m^2\lambda k(2m)^{d+1}.$$

Lemma 6 and Lemma 7 show that when

$$\tau = O(\|\mathbf{R}\hat{\mathbf{H}}_{msk\to r}\hat{\mathbf{H}}_{r,r}\|_\infty) = O(\lambda, k, (2m)^{d+3}),$$

it is efficient to compute

$$\mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}} \leftarrow \mathsf{RandBasis}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathsf{ExtendLeft}(\mathbf{B}, \mathbf{G}, \mathbf{T}_\mathbf{G}, \mathbf{R}\hat{\mathbf{H}}_{msk\to r}\hat{\mathbf{H}}_{r,r}), \tau).$$

Then the challenger runs $\mathsf{SamplePre}$ algorithm as in $\mathbf{Game}_4$. Besides, the challenger calls KeyGen_2 (msk, x) algorithm when $\mathcal{A}$ makes a re-encryption query on $((x, g), (f, k))$ such that $k \notin \mathsf{Deriv}$. Since the responses to key queries and re-encryption queries are statistically close to those in the previous game, the adversary's advantage is at most negligibly different from its advantage in $\mathbf{Game}_4$. Therefore, we have $|\Pr[S_4] - \Pr[S_5]| \leq \mathsf{negl}(\lambda)$.

$\mathbf{Game}_6$: In this game, we change the way the matrix $\mathbf{B}$ is generated. Concretely, the challenger chooses $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n\times m'}$ without producing the corresponding trapdoor $\mathbf{T}_\mathbf{B}$. By Lemma 1, this makes only $2^{-\Omega(n)}$-statistical distance with uniform distribution. The challenger could answer all key queries without the trapdoor because of the change we made in $\mathbf{Game}_5$, the view of $\mathcal{A}$ is altered only negligibly. Therefore, we have $|\Pr[S_5] - \Pr[S_6]| \leq \mathsf{negl}(\lambda)$.

$\mathbf{Game}_7$: In this game, we change the way the challenge ciphertext is created. The challenger chooses $(\mathbf{u}_0^*, \mathbf{u}_1^*, u_2^*) \in \mathbb{Z}_q^{1\times(m'+ml_f+1)}$ at random. Since the challenge ciphertext completely hides b, thus $\mathcal{A}$ has no advantage in this game. We claim that $|\Pr[S_6] - \Pr[S_7]|$ is negligible for a PPT adversary assuming the hardness of DLWE problem. We show this by giving a reduction from DLWE problem.

Reduction from LWE. Suppose $\mathcal{A}_1$ has a non-negligible advantage in distinguish $\mathbf{Game}_6$ and $\mathbf{Game}_7$. We use $\mathcal{A}_1$ to construct an LWE adversary $\mathcal{B}$ as follows:

LWE Instance. $\mathcal{B}$ receives an LWE instance as $(\mathbf{B}\|\mathbf{v}, \mathbf{u}_0\|u_2') \in \mathbb{Z}_q^{n\times(m'+1)} \times \mathbb{Z}_q^{1\times(m'+1)}$. The task of $\mathcal{B}$ is to distinguish whether $\mathbf{u}_0\|u_2' = \mathbf{s}^T(\mathbf{B}\|\mathbf{v}) + \bar{\mathbf{e}}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ and $\bar{\mathbf{e}} \in \chi^{m'+1}$ or $\mathbf{u}_0\|u_2' \xleftarrow{\$} \mathbb{Z}_q^{1\times(m'+1)}$.

Phase 1 (Setup): $\mathcal{B}$ sets $\mathbf{B}$ and $\mathbf{v}$ to be LWE terms. Unlike the real scheme, $\mathcal{B}$ does not require to generate the trapdoor $\mathbf{T}_\mathbf{B}$ of the matrix $\mathbf{B}$ (*i.e.*, the change we made in $\mathbf{Game}_6$). It assembles public parameters *pp* as in the previous game: compute $(P.pp, P.msk) \leftarrow \mathsf{P.Setup}(1^\lambda)$, set $\sigma = P.msk$ and $\mathbf{A} = \mathbf{BR} + \sigma \otimes \mathbf{G}$, where $\mathbf{R} \xleftarrow{\$} \{1, -1\}^{m'\times m\lambda}$. $\mathcal{B}$ gives $\mathcal{A}_1$ the public parameters

$$pp = (\mathbf{B}, \mathbf{A}, \mathbf{v}, P.pp).$$

The master secret key contains only σ. Then, it initializes a counter $\mathsf{numCt} := 0$, a policy-value store $\mathsf{C} := \emptyset$, a key list $\mathsf{K} := \emptyset$ and a set $\mathsf{Deriv} := \emptyset$.

Phase 2 (Oracle Query): $\mathcal{B}$ answers $\mathcal{A}_1$'s key queries, encryption queries, re-encryption key queries and re-encryption queries as in $\mathbf{Game}_7$, except for challenge query.

- $\mathcal{O}_{\mathsf{Cha}}(f^*, (\mu_0, \mu_1))$: To generate the challenge ciphertext, $\mathcal{B}$ first picks a random bit $b \in \{0, 1\}$, computes $s_{f^*}, \mathbf{u}_1^*$ as in $\mathbf{Game}_7$ and outputs $ct_{f^*} = (s_{f^*}, \mathbf{u}_0^* = \mathbf{u}_0, \mathbf{u}_1^*, \mathbf{u}_2^* = u_2' + \mu_b \lfloor q/2 \rceil)$). Then it sets $\mathsf{numCt} := \mathsf{numCt} + 1$ and $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. $\mathcal{B}$ adds ct_{f^*} in C with policy tuple (f^*, numCt) and gives $(\mathsf{numCt}, ct_{f^*})$ to $\mathcal{A}_1$.

Phase 3 (Decision): At the end of the game, $\mathcal{A}_1$ guesses if it is interacting with the challenger of $\mathbf{Game}_6$ or $\mathbf{Game}_7$. $\mathcal{B}$ outputs $\mathcal{A}_1$'s guess as the answer to the LWE challenge it is trying to distinguish.

It can be seen that if $(\mathbf{B}\|\mathbf{v}, \mathbf{u}_0\|u_2')$ is a valid LWE instance (*i.e.*, $(\mathbf{u}_0\|u_2') = \mathbf{s}^T(\mathbf{B}\|\mathbf{v}) + \bar{\mathbf{e}}$), the view of the adversary corresponds to $\mathbf{Game}_6$. Otherwise (*i.e.*, $(\mathbf{u}_0\|u_2') \xleftarrow{\$} \mathbb{Z}_q^{1\times(m'+1)}$), it corresponds to $\mathbf{Game}_7$. Besides, observe that $\mathbf{u}_1^* = \mathbf{u}_0\mathbf{R}\hat{\mathbf{H}}_{msk\to s_{f^*}} + \bar{\mathbf{e}}_1^T$ is uniform and independent in $\mathbb{Z}_q^{1\times ml_f}$ by a standard application of the leftover hash lemma (see Definition 10). We therefore conclude that supposing the hardness of DLWE problem, we have $|\Pr[S_6] - \Pr[S_7]| \leq \mathsf{negl}(\lambda)$.

Therefore, combing the above conclusions together, the theorem is proven. □

5 Adaptively HRA-Secure CAB-PRE from LWE

In this section, we present an adaptively HRA-secure CAB-PRE scheme based on the AB-PRE construction outlined above, in which the condition supports inner-product predicates.

5.1 Technique Overview

We augment the AB-PRE mentioned above by utilizing a delegation condition constructed from a cPRF for inner-product predicates [10] to yield a unidirectional, single-hop CAB-PRE scheme. In this scheme, the decryption procedure remains virtually unchanged between the two levels of ciphertexts, with only minor restrictions on the noise bound. Notably, the cPRF for inner-product predicates [10] possesses expressibility between t-CNF and NC^1 and shares a similar intuition with the delegation mechanism we aim to construct. While it is feasible to integrate inner-product predicates and key switching techniques into the scheme, it may require an expansion of dimensionality in generating re-encryption keys.

We employ the inner-product predicate $C_{\boldsymbol{\beta}}$ in our CAB-PRE scheme and process the predicate vector using vector decomposition defined in Definition 9. The usage of vector decomposition helps prevent noise explosions. We insert inner-product vectors in the second-level encryption and re-encryption key phases to ensure that only ciphertexts satisfying a specific condition can be re-encrypted by the proxy. Moreover, the second-level encryption algorithm provides a weak form of anonymity, *i.e.*, attribute hiding, by embedding the vector $\boldsymbol{\alpha}$ into the ciphertext component $\mathbf{u}_3 = \mathbf{s}^T(\mathbf{D} + \mathbf{h} \otimes \mathsf{P2}(\boldsymbol{\alpha})^T) + \mathbf{e}_3^T$. It should note that $\boldsymbol{\alpha}$ remains hidden even with knowledge of the ciphertext, as $\mathbf{s}$ and $\mathbf{e}_3$ are confidential, $\mathbf{D}$ and $\mathbf{h}$ are uniformly random. Then, we use the $\mathsf{SamplePre}$ algorithm to sample a short vector $\mathbf{d}$ such that $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{d} = \mathbf{v}'$, where $\mathbf{v}' = (\mathbf{V} + \mathbf{D}) \cdot \mathsf{BD}(\boldsymbol{\beta})$. The re-encryption key, denoted as

$$\mathbf{Z} = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{V} + \mathbf{R}_3 - \mathsf{P2}(\mathbf{d}) \otimes \boldsymbol{\gamma} \\ \mathbf{0}_{l\lceil \log q\rceil \times (m'+ml_g)} & \mathsf{BD}(\boldsymbol{\beta}) \otimes \boldsymbol{\gamma} \end{pmatrix},$$

is captured, with $\boldsymbol{\gamma} \xleftarrow{\$} \{0,1\}^{l\lceil \log q\rceil}$.

The proof idea of HRA security for CAB-PRE is the same as AB-PRE; ultimately, it is all reducible to the decisional LWE problem. However, the CAB-PRE scheme adds a ciphertext component $\mathbf{u}_3$, so we need to perform additional analysis. For this challenge ciphertext component in the proof, we use the lossy model for LWE to show that it is indistinguishable from the uniform distribution.

5.2 Construction

Let $C_{\boldsymbol{\beta}} : \mathbb{Z}_q^l \to \{0,1\}$ be an inner-product predicate [5] with dimension l, which evaluates $C_{\boldsymbol{\beta}}(\boldsymbol{\alpha}) = 1$ when given an input $\boldsymbol{\alpha} \in \mathbb{Z}_q^l$ such that $\langle \boldsymbol{\beta}, \boldsymbol{\alpha} \rangle = 0$. We define an adaptively HRA-secure CAB-PRE for inner-product as follows.

$\mathsf{Setup}(1^\lambda)$: Identical to AB-PRE except that: replace $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$ with $\mathbf{V} \xleftarrow{\$} \mathbb{Z}_q^{n \times l\lceil \log q\rceil}$ and sample additionally a matrix $\mathbf{D} \xleftarrow{\$} \mathbb{Z}_q^{n \times l\lceil \log q\rceil}$. Output $pp = (\mathbf{B}, \mathbf{A}, \mathbf{V}, \mathbf{D}, P.pp)$ and $msk = (\sigma, \mathbf{T_B})$.

$\mathsf{KeyGen}(msk, x)$: Identical to AB-PRE except that: sample $\mathbf{K} \leftarrow \mathsf{SamplePre}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{V}, \tau)$ such that $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{K} = \mathbf{V}$. Output $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{K})$.

$\mathsf{Enc}_1(f, \mu)$**:** The way to generate s_f, $\mathbf{u}_0$ and $\mathbf{u}_1$ is identical to AB-PRE except that: choose randomly an error $\mathbf{e}_2 \xleftarrow{\$} \chi^{l\lceil \log q\rceil}$, compute

$$\mathbf{u}_2 = \mathbf{s}^T\mathbf{V} + \mathbf{e}_2^T + \mu(\mathbf{0}\|\mathbf{g}^T),$$

where the zero vector has dimension $(l-1)\lceil \log q\rceil$. Output $ct_f^1 = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2)$.

[5] We require that the predicate vector $\boldsymbol{\beta} = (\beta_1, ..., \beta_l) \in \mathbb{Z}_q^l$ satisfies a special requirement, *i.e.*, $\beta_l \neq 0$, to better define the noise bound and ensure the decryption correct. Without loss of generality, we assume that of $\beta_l = q - 1$ in decryption phase.

$\mathsf{Enc}_2(f, \boldsymbol{\alpha}, \mu)$**:** Choose a vector $\mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^n$ and an error $\mathbf{e}_3 \xleftarrow{\$} \chi^{l\lceil \log q \rceil}$. Identical to Enc_1 except that adding a component

$$\mathbf{u}_3 = \mathbf{s}^T(\mathbf{D} + \mathbf{h} \otimes \mathsf{P2}(\boldsymbol{\alpha})^T) + \mathbf{e}_3^T.$$

Output $ct_f^2 = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$.

$\mathsf{Dec}(sk_x, ct_f)$**:** Parse $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{K})$ and $ct_f \in \{ct_f^1, ct_f^2\}$. Evaluate $\hat{\mathbf{H}}_{r,r'}$ and $\hat{\mathbf{H}}_{s_f \to r'}$ identical to AB-PRE. Then compute

$$\mu = \left\lceil \mathbf{u}_2 - (\mathbf{u}_0 \| \mathbf{u}_1 \hat{\mathbf{H}}_{s_f \to r'} \hat{\mathbf{H}}_{r,r'})\mathbf{K} \right\rfloor_2,$$

in which $\lceil \cdot \rfloor_2 : \mathbb{Z}_q \to \{0,1\}$ indicates its penultimate is closer modulo q to 0 or to a certain upper bound. It is noteworthy that the decryption algorithm remains constant regardless of the level of ciphertexts, *i.e.*, first or second. Decrypting the original ciphertext is subject to an upper limit of $2^{\lceil \log q \rceil - 2}$. However, when decrypting a transformed ciphertext, a coefficient factor linked to the predicate vector $\boldsymbol{\beta}$ is involved. In such cases, adjusting the noise bound appropriately ensures the accurate retrieval of the message μ.

$\mathsf{ReKeyGen}(sk_x, g, \boldsymbol{\beta})$**:** Parse $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{K})$. Identical to AB-PRE except that: for a predicate vector $\boldsymbol{\beta} \in \mathbb{Z}_q^l$, let $\mathbf{v}' = (\mathbf{V} + \mathbf{D}) \cdot \mathsf{BD}(\boldsymbol{\beta})$. Sample $\mathbf{d} \leftarrow \mathsf{SamplePre}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{v}', \tau)$ such that $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{d} = \mathbf{v}'$. Select randomly $\mathbf{R}_1 \in \chi^{(m'+m)\lceil \log q \rceil \times n}$, $\mathbf{R}_2 \in \chi^{(m'+m)\lceil \log q \rceil \times (m'+ml_g)}$, $\mathbf{R}_3 \in \chi^{(m'+m)\lceil \log q \rceil \times l \lceil \log q \rceil}$, and $\gamma \in \{0,1\}^{l\lceil \log q \rceil}$, set

$$\mathbf{Z} = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{V} + \mathbf{R}_3 - \mathsf{P2}(\mathbf{d}) \otimes \gamma \\ \mathbf{0}_{l\lceil \log q \rceil \times (m'+ml_g)} & \mathsf{BD}(\boldsymbol{\beta}) \otimes \gamma \end{pmatrix}.$$

Output $rk_{x \to g}^{\boldsymbol{\beta}} = (s_g, r, \boldsymbol{\beta}, \gamma, \mathbf{Z})$.

$\mathsf{ReEnc}(rk_{x\to g}^{\boldsymbol{\beta}}, ct_f^2)$**:** Parse $rk_{x\to g}^{\boldsymbol{\beta}} = (s_g, r, \boldsymbol{\beta}, \gamma, \mathbf{Z})$ and $ct_f^2 = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$. Identical to AB-PRE except that: let $\mathbf{u}_1' = \mathbf{u}_1 \hat{\mathbf{H}}_{s_f \to r'} \hat{\mathbf{H}}_{r,r'}$ and $\mathbf{u}_2' = \mathbf{u}_2 + \mathbf{u}_3$, evaluate

$$ct_{f \to g} = (\mathsf{BD}(\mathbf{u}_0 \| \mathbf{u}_1')\|\mathbf{u}_2') \cdot \mathbf{Z}.$$

Output $ct_g^1 = (s_g, \boldsymbol{\beta}, \gamma, ct_{f \to g})$.

5.3 Correctness and Choice of Parameters

Theorem 4. *The CAB-PRE scheme is correct with respect to f and $C_{\boldsymbol{\beta}}$ under proper parameters.*

Proof. The correct decryption can be bifurcated into two components: the original ciphertext of either the first-level or second-level, as they undergo similar decryption operations, and the re-encrypted ciphertext.

First, for the security parameter λ, given functions $f \in \mathcal{F}$ and I_r, two strings $x \in \{0,1\}^l$ and $r' \in \{0,1\}^k$ such that $f(x) = 1 \wedge I_r(r') = 0$, if $(pp, msk) \leftarrow \mathsf{Setup}(1^\lambda)$, $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$, $ct_f^1 \leftarrow \mathsf{Enc}_1(f, \mu)$ and $ct_f^2 \leftarrow \mathsf{Enc}_2(f, \boldsymbol{\alpha}, \mu)$, then we have $\mu \leftarrow \mathsf{Dec}(sk_x, ct_f \in \{ct_f^1, ct_f^2\})$ with a non-negligible probability. The detailed decryption and choice of parameters are similar to AB-PRE: computing $\mathbf{A}_x, \mathbf{A}_f, (\mathbf{H}_r, \hat{\mathbf{H}}_{r,r'})$ and $(\mathbf{H}_{f\to x}, \hat{\mathbf{H}}_{s_f \to r'})$, we have

$$[\mathbf{A}_f - s_f \otimes \mathbf{G}]\hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'} = \mathbf{A}_{x,r}$$

and

$$\mathbf{u}_2 - (\mathbf{u}_0 \| \mathbf{u}_1 \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'})\mathbf{K} = \mu(\mathbf{0}\|\mathbf{g}^T) + \mathbf{e}_2^T - (\mathbf{e}_0^T \| \mathbf{e}_1')\mathbf{K},$$

where $\mathbf{e}_1' = \mathbf{e}_1^T \hat{\mathbf{H}}_{s_f \to r'}\hat{\mathbf{H}}_{r,r'}$ and $\|\mathbf{e}_1'\|_\infty \le m^2 l_f k \tilde{B}(2m)^{d_{\mathsf{ConEval}}+1}$.

By applying the property of tailcut inequality (see Definition 4) on matrices, we get

$$\|\mathbf{K}\|_\infty \le \tau\sqrt{(m'+m)\cdot l\lceil \log q \rceil}.$$

Therefore, if $l \in O(n)$, $m', l_f, k \in O(n, \lceil \log q \rceil)$, $\tilde{B} \in O(B, n)$ and $\tau \in O(\lambda, k, (2m)^{d+3})$, then

$$\begin{aligned}\|\mathbf{e}_2^T - (\mathbf{e}_0^T\|\mathbf{e}_1')\mathbf{K}\|_\infty &\le \|\mathbf{e}_2^T\|_\infty + (m'\|\mathbf{e}_0^T\|_\infty + m\|\mathbf{e}_1'\|_\infty)\|\mathbf{K}\|_\infty \\ &\le l\lceil \log q \rceil B + (m'B + m^3 l_f k \tilde{B}(2m)^{d_{\mathsf{ConEval}}+1})\tau\sqrt{(m'+m)\cdot l\lceil \log q \rceil} \\ &\le B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}.\end{aligned}$$

To capture the correct decryption, the magnitude of penultimate coordinate should be less than $q/8$, namely

$$\|\mathbf{e}_2^T - (\mathbf{e}_0^T\|\mathbf{e}_1')\mathbf{k}\|_\infty \le B \cdot \mathsf{poly}(n\lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4} < q/8,$$

which is overwhelming. The decryption of the original ciphertext is correct.

Now it remains to show how to guarantee the correctness of the re-encrypted ciphertext. Given a re-encryption key, a second-level ciphertext associated with function f can be efficiently re-encrypted to a first-level ciphertext associated with function g, where the delegation condition is described as the inner-product predicates between two vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ over $\mathbb{Z}_q^l$. Specifically, parse the re-encryption key $rk_{x\to g}^{\boldsymbol{\beta}} = (s_g, r, \boldsymbol{\beta}, \gamma, \mathbf{Z})$ and the second-level ciphertext $ct_f^2 = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$, the re-encryption process in the following way:

If $f(x) = 1 \wedge I_r(r') = 0$ holds, where $r' \leftarrow U_{f\to x}(s_f)$, we compute

$$\mathbf{u}_1' = \mathbf{u}_1\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'} = \mathbf{s}^T\mathbf{A}_{x,r} + \mathbf{e}_1^T\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'}$$

and

$$\mathbf{u}_2' = \mathbf{u}_2 + \mathbf{u}_3 = \mathbf{s}^T(\mathbf{V} + \mathbf{D} + \mathbf{h} \otimes \mathsf{P2}(\boldsymbol{\alpha})^T) + \bar{\mathbf{e}}^T + \mu(\mathbf{0}\|\mathbf{g}^T),$$

where $\bar{\mathbf{e}} = \mathbf{e}_2 + \mathbf{e}_3$. If $l = (l_g+1)n + \lambda + 1$ and $C_{\boldsymbol{\beta}}(\boldsymbol{\alpha}) = 1$, we have

$$\begin{aligned} ct_{f\to g} &= (\mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\|\mathbf{u}_2') \cdot \mathbf{Z} \\ &= (\mathbf{c}_0\|\mathbf{c}_1 \;\; \mathbf{c}_2).\end{aligned}$$

Let $\bar{\mathbf{s}} = (\mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\mathbf{R}_1)^T$, then

$$\begin{aligned}\mathbf{c}_0\|\mathbf{c}_1 &= \bar{\mathbf{s}}^T(\mathbf{B}\|[\mathbf{A}_g - s_g \otimes \mathbf{G}]) + \mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\mathbf{R}_2 \\ &= \bar{\mathbf{s}}^T(\mathbf{B}\|[\mathbf{A}_g - s_g \otimes \mathbf{G}]) + (\bar{\mathbf{e}}_0^T\|\bar{\mathbf{e}}_1^T)\end{aligned}$$

and

$$\mathbf{c}_2 = \bar{\mathbf{s}}^T\mathbf{V} + \bar{\mathbf{e}}_2 + \mu(\mathbf{0}\|\mathbf{g}^T) \cdot \mathsf{BD}(\boldsymbol{\beta}) \otimes \gamma,$$

where $\bar{\mathbf{e}}_2 = \mathsf{BD}(\mathbf{u}_0\|\mathbf{u}_1')\mathbf{R}_3 - ((\mathbf{e}_0^T\|\mathbf{e}_1')\mathbf{d} - \bar{\mathbf{e}}^T \cdot \mathsf{BD}(\boldsymbol{\beta})) \otimes \gamma$ and $\|\bar{\mathbf{e}}_2\|_\infty \le B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}$.

Let $\boldsymbol{\beta} = (\beta_1, ..., \beta_l) \in \mathbb{Z}_q^l$, it holds that $(\mathbf{0}\|\mathbf{g}^T) \cdot \mathsf{BD}(\boldsymbol{\beta}) \neq 0$ (where $\beta_l = q-1$) with a significant probability. Thus, we get

$$(\mathbf{0}\|\mathbf{g}^T) \cdot \mathsf{BD}(\boldsymbol{\beta}) \otimes \gamma = (\sum_{i=0}^{\lceil \log q \rceil - 1} \beta_{l,i} \cdot 2^i) \otimes \gamma,$$

where $\beta_{l,i} \in \{0, 1\}$ denotes the binary decomposition of β_l. Without loss of generality, let $sk_y = (\bar{r}, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{y,\bar{r}}}, \bar{\mathbf{K}})$ denote a secret key for an attribute string y that satisfies $g(y) = 1$ and $I_{\bar{r}}(\bar{r}') = 0$, where $\bar{r}' \leftarrow U_{g\to y}(s_g)$. Suppose that the penultimate bit of γ is 1, to capture the correct decryption, we compute

$$\mu = \left\lceil \mathbf{c}_2 - (\mathbf{c}_0\|\mathbf{c}_1\hat{\mathbf{H}}_{s_g\to\bar{r}'}\hat{\mathbf{H}}_{\bar{r},\bar{r}'})\bar{\mathbf{K}} \right\rfloor_2,$$

here $\lceil\cdot\rfloor_2$ denotes whether its penultimate is closer modulo q to 0 or to $\sum_{i=0}^{\lceil \log q \rceil - 1} \beta_{l,i} \cdot 2^i$.

Based on our parameter setting (that is identical to Section 4.3) and analysis, the norm of error term is bounded by

$$\|\bar{\mathbf{e}}_2^T - (\bar{\mathbf{e}}_0^T\|\bar{\mathbf{e}}_1')\bar{\mathbf{K}}\|_\infty \le 2B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4} < \sum_{i=0}^{\lceil \log q \rceil - 1} \beta_{l,i} \cdot 2^{i+1}$$

in which $\bar{\mathbf{e}}_1' = \bar{\mathbf{e}}_1^T\hat{\mathbf{H}}_{s_g\to\bar{r}'}\hat{\mathbf{H}}_{\bar{r},\bar{r}'}$ with an overwhelming probability. The decryption of the re-encrypted ciphertext is correct. □

5.4 Security Proof

Now we give the security proof of CAB-PRE scheme, which involves two algorithms KeyGen_1 and KeyGen_2 defined in Section 4.4.

Theorem 5. *For a class family $\mathcal{F}$ and an inner-product predicate $C_{\boldsymbol{\beta}}$, if P is a conforming cPRF, then CAB-PRE is a unidirectional, single-hop, adaptively HRA-secure scheme under the hardness of DLWE$_{n,q,\chi}$ problem.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the adaptive HRA security of CAB-PRE. We show that the proof proceeds in a sequence of games. In each game, we define S_i to be the event that $\mathcal{A}$ wins in $\mathbf{Game}_i$.

$\mathbf{Game}_0$: This is the original security game from Definition 13. Similar to $\mathbf{Game}_0$ in Section 4.4, we modify the answers of re-encryption key queries as follows:

– When $\mathcal{A}$ makes a re-encryption key query on $(x, g, \boldsymbol{\beta})$, there exist an attribute string y such that $g(y) = 0$. The challenger chooses two matrices $\mathbf{Z}_1 \xleftarrow{\$} \chi^{(m'+m)\lceil \log q\rceil \times (m'+ml_g)}$ and $\mathbf{Z}_2 \xleftarrow{\$} \chi^{(m'+m)\lceil \log q\rceil \times l\lceil \log q\rceil}$, simulates the fifth entries of the re-encryption key as

$$\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_1 & \mathbf{Z}_2 \\ \mathbf{0}_{l\lceil \log q\rceil \times (m'+ml_g)} & \mathsf{BD}(\boldsymbol{\beta}) \otimes \gamma \end{pmatrix},$$

where $\gamma \xleftarrow{\$} \{0,1\}^{l\lceil \log q\rceil}$. It returns $rk^{\boldsymbol{\beta}}_{x\to g}$ to $\mathcal{A}$.

– When $\mathcal{A}$ makes a re-encryption key query on $(x, g, \boldsymbol{\beta})$, there exist an attribute string y such that $g(y) = 1$. In addition to the termination conditions described in Definition 13, the challenger first invokes $\mathsf{KeyGen}_1(msk, x)$ algorithm (here $msk = (\sigma, \mathbf{T_B})$) to obtain $\hat{sk}_x (= \mathbf{d})$ without generating the secret key sk_x. It then computes $rk^{\boldsymbol{\beta}}_{x\to g} \leftarrow \mathsf{ReKeyGen}(\hat{sk}_x, g, \boldsymbol{\beta})$ and gives $rk^{\boldsymbol{\beta}}_{x\to g}$ to $\mathcal{A}$.

By Definition 13 and Definition 14, we have $|\Pr[S_0] - 1/2| \le \mathsf{negl}(\lambda)$.

Remark 4. Recall in the real scheme, the challenger creates

$$\mathbf{Z} = \begin{pmatrix} \mathbf{R}_1\mathbf{H} + \mathbf{R}_2 & \mathbf{R}_1\mathbf{V} + \mathbf{R}_3 - \mathsf{P2}(\mathbf{d}) \otimes \gamma \\ \mathbf{0}_{l\lceil \log q\rceil \times (m'+ml_g)} & \mathsf{BD}(\boldsymbol{\beta}) \otimes \gamma \end{pmatrix},$$

where $\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3$ are sampled from discrete Gaussian distributions. Observed that $\mathbf{Z}$ is indistinguishable with uniform distribution based on the hardness of HNF-LWE problem.

$\mathbf{Game}_1$: Identical to $\mathbf{Game}_1$ in Section 4.4.

$\mathbf{Game}_2$: Identical to $\mathbf{Game}_2$ in Section 4.4 except that the adversary submits a challenge tuple $(f^*, \boldsymbol{\alpha}^*, (\mu_0, \mu_1))$ in challenge phase.

$\mathbf{Game}_3$: Identical to $\mathbf{Game}_3$ in Section 4.4.

$\mathbf{Game}_4$: Identical to $\mathbf{Game}_4$ in Section 4.4, except that: the way the challenger generates $\mathbf{u}_3^*$ (if $\boldsymbol{\alpha}^* \neq \mathsf{Null}$) is also unaltered, for the challenge query $(f^*, \boldsymbol{\alpha}^*, (\mu_0, \mu_1))$ that the adversary makes.

$\mathbf{Game}_5$: Identical to $\mathbf{Game}_5$ in Section 4.4, except that: the challenger calls $\mathsf{KeyGen}_2(msk, x)$ algorithm when $\mathcal{A}$ makes a re-encryption query on $((x, g, \boldsymbol{\beta}), (f, \boldsymbol{\alpha}, k))$ such that $k \notin \mathsf{Deriv}$.

$\mathbf{Game}_6$: Identical to $\mathbf{Game}_6$ in Section 4.4.

$\mathbf{Game}_7$: In this game, we change the way the challenge ciphertext is generated. The challenger chooses $(\mathbf{u}_0^*, \mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*) \in \mathbb{Z}_q^{1\times(m'+ml_f+2l\lceil \log q\rceil)}$ (if $\boldsymbol{\alpha}^* \neq \mathsf{Null}$) at random. Since the challenge ciphertext completely hides b, thus $\mathcal{A}$ has no advantage in this game. We claim that $|\Pr[S_6] - \Pr[S_7]|$ is negligible for a PPT adversary assuming the hardness of DLWE problem. To show that, we do by giving a reduction from DLWE problem.

Reduction from LWE. Suppose $\mathcal{A}_2$ has a non-negligible advantage in distinguish $\mathbf{Game}_6$ and $\mathbf{Game}_7$. We use $\mathcal{A}_2$ to construct an LWE adversary $\mathcal{B}$ as follows:

LWE Instance. $\mathcal{B}$ receives an LWE instance as $(\mathbf{B}\|\mathbf{V}, \mathbf{u}_0\|\mathbf{u}_2') \in \mathbb{Z}_q^{n\times(m'+l\lceil\log q\rceil)} \times \mathbb{Z}_q^{1\times(m'+l\lceil\log q\rceil)}$. The task of $\mathcal{B}$ is to distinguish whether $\mathbf{u}_0\|\mathbf{u}_2' = \mathbf{s}^T(\mathbf{B}\|\mathbf{V}) + \bar{\mathbf{e}}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ and $\bar{\mathbf{e}} \in \chi^{m'+l\lceil\log q\rceil}$ or $\mathbf{u}_0\|\mathbf{u}_2' \xleftarrow{\$} \mathbb{Z}_q^{1\times(m'+l\lceil\log q\rceil)}$.

Phase 1 (Setup): $\mathcal{B}$ sets $\mathbf{B}$ and $\mathbf{V}$ to be LWE matrices. It computes $(P.pp, P.msk) \leftarrow \mathsf{P.Setup}(1^\lambda)$, sets $\sigma = P.msk$ and $\mathbf{A} = \mathbf{BR} + \sigma \otimes \mathbf{G}$, where $\mathbf{R} \xleftarrow{\$} \{1, -1\}^{m'\times m\lambda}$. $\mathcal{B}$ gives $\mathcal{A}_2$ the public parameters

$$pp = (\mathbf{B}, \mathbf{A}, \mathbf{V}, \mathbf{D}, P.pp),$$

where $\mathbf{D} \xleftarrow{\$} \mathbb{Z}_q^{n\times l\lceil\log q\rceil}$. The master secret key contains only σ. Then, $\mathcal{B}$ initializes a counter $\mathsf{numCt} := 0$, a policy-value store $\mathsf{C} := \emptyset$, a key list $\mathsf{K} := \emptyset$ and a set $\mathsf{Deriv} := \emptyset$.

Phase 2 (Oracle Query): $\mathcal{B}$ answers $\mathcal{A}_2$'s key queries, encryption queries, re-encryption key queries and re-encryption queries as in $\mathbf{Game}_7$, except that the challenge query.

– $\mathcal{O}_{\mathsf{Cha}}(f^*, \boldsymbol{\alpha}^*, (\mu_0, \mu_1))$: To generate the challenge ciphertext, $\mathcal{B}$ first picks a random bit $b \in \{0, 1\}$, computes $s_{f^*}, \mathbf{u}_1^*, \mathbf{u}_3^*$ as in $\mathbf{Game}_7$ and outputs $ct_{f^*} = (s_{f^*}, \mathbf{u}_0^* = \mathbf{u}_0, \mathbf{u}_1^*, \mathbf{u}_2^* = \mathbf{u}_2' + \mu_b(\mathbf{0}\|\mathbf{g}^T), \mathbf{u}_3^*)$ (if $\boldsymbol{\alpha}^* \neq \mathsf{Null}$). Then it sets $\mathsf{numCt} := \mathsf{numCt} + 1$ and $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. $\mathcal{B}$ adds ct_{f^*} in C with policy tuple $(f^*, \boldsymbol{\alpha}^*, \mathsf{numCt})$ and gives $(\mathsf{numCt}, ct_{f^*})$ to $\mathcal{A}_2$.

Phase 3 (Decision): Finally, $\mathcal{A}_2$ guesses if it is interacting with the challenger of $\mathbf{Game}_6$ or $\mathbf{Game}_7$. $\mathcal{B}$ outputs $\mathcal{A}_2$'s guess as the answer to the LWE challenge it is trying to distinguish.

It is obvious that if $(\mathbf{B}\|\mathbf{V}, \mathbf{u}_0\|\mathbf{u}_2')$ is a valid LWE instance (*i.e.*, $(\mathbf{u}_0\|\mathbf{u}_2') = \mathbf{s}^T(\mathbf{B}\|\mathbf{V}) + \bar{\mathbf{e}}$), the view of the adversary corresponds to $\mathbf{Game}_6$. Otherwise (*i.e.*, $(\mathbf{u}_0\|\mathbf{u}_2') \xleftarrow{\$} \mathbb{Z}_q^{1\times(m'+l\lceil\log q\rceil)}$), it corresponds to $\mathbf{Game}_7$. Besides, observe that $\mathbf{u}_1^* = \mathbf{u}_0\mathbf{R}\hat{\mathbf{H}}_{msk\to s_{f^*}} + \bar{\mathbf{e}}_1^T$ is uniform and independent in $\mathbb{Z}_q^{1\times ml_f}$ by a standard application of the leftover hash lemma (see Definition 10). Moreover, for any condition vector $\boldsymbol{\alpha}^* \in \mathbb{Z}_q^l$, we have $\mathbf{u}_3^* = \mathbf{s}^T(\mathbf{D} + \mathbf{h} \otimes \mathsf{P2}(\boldsymbol{\alpha}^*)^T) + \bar{\mathbf{e}}_3^T$ which can be regarded as the lossy mode for LWE (see Definition 8) and is close to uniformly random. We use $\mathsf{SampleLossy}$ to describe the procedure that sample a matrix in the lossy mode. Let n, l be positive integers, and $\boldsymbol{\alpha}$ be a vector over $\mathbb{Z}_q^l$.

$\mathsf{SampleLossy}(n, l, \boldsymbol{\alpha})$: It samples $\mathbf{D} \xleftarrow{\$} \mathbb{Z}_q^{n\times l\lceil\log q\rceil}$ and $\mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^n$, outputs $\mathbf{A}_1 = \mathbf{D} + \mathbf{h} \otimes \mathsf{P2}(\boldsymbol{\alpha})^T$.

It is easy to see that $\mathbf{A}_1$ in the lossy mode is within a negligible statistical distance from uniform distribution. Choosing $\mathbf{A}_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times l\lceil\log q\rceil}$ and $\mathbf{A}_1 \leftarrow \mathsf{SampleLossy}(n, l, \boldsymbol{\alpha})$, we know that $\mathbf{A}_0$ and $\mathbf{A}_1$ are computationally indistinguishable, denoted by $\mathbf{A}_0 \overset{\mathrm{c}}{\approx} \mathbf{A}_1$. Then for $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e} \xleftarrow{\$} \chi^{l\lceil\log q\rceil}$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{l\lceil\log q\rceil}$, it holds that

$$(\mathbf{A}_0, \mathbf{s}^T\mathbf{A}_0 + \mathbf{e}^T) \overset{\mathrm{c}}{\approx} (\mathbf{A}_1, \mathbf{s}^T\mathbf{A}_1 + \mathbf{e}^T).$$

On the other hand, we claim that

$$(\mathbf{A}_0, \mathbf{s}^T\mathbf{A}_0 + \mathbf{e}^T) \overset{\mathrm{c}}{\approx} (\mathbf{A}_0, \mathbf{u}^T)$$

under the DLWE problem. Finally, we have

$$(\mathbf{A}_1, \mathbf{s}^T\mathbf{A}_1 + \mathbf{e}^T) \overset{\mathrm{c}}{\approx} (\mathbf{A}_1, \mathbf{u}^T).$$

In other words, the ciphertext component $\mathbf{u}_3^*$ is computationally indistinguishable from the uniform distribution over $\mathbb{Z}_q^{l\lceil\log q\rceil}$. We therefore conclude that supposing the hardness of DLWE problem, we have $|\Pr[S_6] - \Pr[S_7]| \leq \mathsf{negl}(\lambda)$.

Therefore, combing the above conclusions together, the theorem is proven. $\square$

6 Conclusion

In this work, we formalize the notion of CAB-PRE and propose the first adaptively HRA-secure CAB-PRE scheme to enrich the PRE application scenarios.

We design the first adaptively HRA-secure (ciphertext-policy) AB-PRE scheme as a building block to construct the CAB-PRE scheme. We highlight that this AB-PRE scheme solves the open problem left by Susilo et al. [30] in ESORICS'21 about constructing an HRA-secure (ciphertext-policy) AB-PRE scheme. Then, we introduce a well-matched conditional delegation mechanism for inner-product predicates based on this AB-PRE scheme to derive our adaptively HRA-secure CAB-PRE scheme. Meanwhile, we provide security proof of these two schemes to confirm their security.

We note that the key-switching technique will incur dimension expansion of the re-encryption key in our construction. Therefore, exploring how to control this dimension expansion will be interesting. In addition, we may require a more robust CAB-PRE scheme in the post-quantum world in some applications, such as a CCA-secure CAB-PRE scheme over lattices. We leave these two directions as future work.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer (2010), https://doi.org/10.1007/978-3-642-13190-5_28
2. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 595–618. Springer (2009), https://doi.org/10.1007/978-3-642-03356-8_35
3. Ateniese, G., Benson, K., Hohenberger, S.: Key-private proxy re-encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 279–294. Springer (2009), https://doi.org/10.1007/978-3-642-00862-7_19
4. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer (1998), https://doi.org/10.1007/BFb0054122
5. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer (2014), https://doi.org/10.1007/978-3-642-55220-5_30
6. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: Goldwasser, S. (ed.) ITCS 2012. pp. 309–325. ACM (2012), https://doi.org/10.1145/2090236.2090262
7. Brakerski, Z., Vaikuntanathan, V.: Circuit-abe from LWE: unbounded attributes and semi-adaptive security. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9816, pp. 363–384. Springer (2016), https://doi.org/10.1007/978-3-662-53015-3_13
8. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer (2010), https://doi.org/10.1007/978-3-642-13190-5_27
9. Cohen, A.: What about bob? the inadequacy of CPA security for proxy reencryption. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 287–316. Springer (2019), https://doi.org/10.1007/978-3-030-17259-6_10
10. Davidson, A., Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Adaptively secure constrained pseudorandom functions in the standard model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12170, pp. 559–589. Springer (2020), https://doi.org/10.1007/978-3-030-56784-2_19
11. Deng, H., Qin, Z., Wu, Q., Guan, Z., Zhou, Y.: Flexible attribute-based proxy re-encryption for efficient data sharing. Information Sciences **511**, 94–113 (2020). https://doi.org/https://doi.org/10.1016/j.ins.2019.09.052
12. Fuchsbauer, G., Kamath, C., Klein, K., Pietrzak, K.: Adaptively secure proxy re-encryption. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 317–346. Springer (2019), https://doi.org/10.1007/978-3-030-17259-6_11
13. Ge, C., Susilo, W., Fang, L., Wang, J., Shi, Y.: A cca-secure key-policy attribute-based proxy re-encryption in the adaptive corruption model for dropbox data sharing system. Des. Codes Cryptogr. **86**(11), 2587–2603 (2018)
14. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) STOC 2008. pp. 197–206. ACM (2008), https://doi.org/10.1145/1374376.1374407
15. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th Annual Symposium on Foundations of Computer Science, West Palm Beach, Florida, USA, 24-26 October 1984. pp. 464–479. IEEE Computer Society (1984). https://doi.org/10.1109/SFCS.1984.715949
16. Huang, Q., Yang, Y., Fu, J.: PRECISE: identity-based private data sharing with conditional proxy re-encryption in online social networks. Future Gener. Comput. Syst. **86**, 1523–1533 (2018). https://doi.org/10.1016/j.future.2017.05.026

17. Katsumata, S., Yamada, S., Yamakawa, T.: Tighter security proofs for GPV-IBE in the quantum random oracle model. In: Peyrin, T., Galbraith, S.D. (eds.) ASIACRYPT 2018. LNCS, vol. 11273, pp. 253–282. Springer (2018). https://doi.org/10.1007/978-3-030-03329-3_9
18. Li, B., Xu, J., Liu, Y.: Lattice-based fuzzy conditional proxy re-encryption. Journal of Internet Technology **20**(5), 1379–1385 (2019)
19. Li, J., Ma, C., Zhang, K.: A novel lattice-based ciphertext-policy attribute-based proxy re-encryption for cloud sharing. In: International Symposium on Security and Privacy in Social Networks and Big Data. pp. 32–46. Springer (2019)
20. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute based proxy re-encryption with delegating capabilities. In: Li, W., Susilo, W., Tupakula, U.K., Safavi-Naini, R., Varadharajan, V. (eds.) ASIACCS 2009. pp. 276–286. ACM (2009), `https://doi.org/10.1145/1533057.1533094`
21. Liang, X., Weng, J., Yang, A., Yao, L., Jiang, Z., Wu, Z.: Attribute-based conditional proxy re-encryption in the standard model under LWE. In: Bertino, E., Shulman, H., Waidner, M. (eds.) ESORICS 2021. LNCS, vol. 12973, pp. 147–168. Springer (2021), `https://doi.org/10.1007/978-3-030-88428-4_8`
22. Luo, F., Al-Kuwari, S., Wang, F., Chen, K.: Attribute-based proxy re-encryption from standard lattices. Theor. Comput. Sci. **865**, 52–62 (2021)
23. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer (2012), `https://doi.org/10.1007/978-3-642-29011-4_41`
24. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: FOCS 2004. pp. 372–381. IEEE Computer Society (2004), `https://doi.org/10.1109/FOCS.2004.72`
25. Pareek, G., Purushothama, B.R.: Proxy visible re-encryption scheme with application to e-mail forwarding. In: Shekhawat, R.S., Gaur, M.S., Elçi, A., Vaidya, J., Makarevich, O.B., Poet, R., Orgun, M.A., Dhaka, V.S., Bohra, M.K., Singh, V., Babenko, L.K., Sheykhkanloo, N.M., Rahnama, B. (eds.) Proceedings of the 10th International Conference on Security of Information and Networks, SIN 2017, Jaipur, IN, India, October 13-15, 2017. pp. 212–217. ACM (2017). https://doi.org/10.1145/3136825.3136906
26. Patil, R.Y.: Digital forensics evidence management based on proxy re-encryption. Int. J. Comput. Appl. Technol. **68**(4), 405–413 (2022). https://doi.org/10.1504/IJCAT.2022.10050318
27. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher, M. (ed.) STOC 2009. pp. 333–342. ACM (2009), `https://doi.org/10.1145/1536414.1536461`
28. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) STOC 2005. pp. 84–93. ACM (2005), `https://doi.org/10.1145/1060590.1060603`
29. Su, M., Zhou, B., Fu, A., Yu, Y., Zhang, G.: Prta: A proxy re-encryption based trusted authorization scheme for nodes on cloudiot. Information Sciences **527**, 533–547 (2020). https://doi.org/https://doi.org/10.1016/j.ins.2019.01.051
30. Susilo, W., Dutta, P., Duong, D.H., Roy, P.S.: Lattice-based hra-secure attribute-based proxy re-encryption in standard model. In: Bertino, E., Shulman, H., Waidner, M. (eds.) ESORICS 2021. LNCS, vol. 12973, pp. 169–191. Springer (2021), `https://doi.org/10.1007/978-3-030-88428-4_9`
31. Tsabary, R.: Fully secure attribute-based encryption for t-cnf from LWE. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 62–85. Springer (2019), `https://doi.org/10.1007/978-3-030-26948-7_3`
32. Weng, J., Deng, R.H., Ding, X., Chu, C., Lai, J.: Conditional proxy re-encryption secure against chosen-ciphertext attack. In: Li, W., Susilo, W., Tupakula, U.K., Safavi-Naini, R., Varadharajan, V. (eds.) ASIACCS 2009. pp. 322–332. ACM (2009), `https://doi.org/10.1145/1533057.1533100`
33. Zhou, Y., Zhao, L., Jin, Y., Li, F.: Backdoor-resistant identity-based proxy re-encryption for cloud-assisted wireless body area networks. Information Sciences **604**, 80–96 (2022). https://doi.org/https://doi.org/10.1016/j.ins.2022.05.007

A Attribute-Based Proxy Re-Encryption

Syntax. A unidirectional, single-hop attribute-based proxy re-encryption (AB-PRE) for policies $\mathcal{F} : \{0,1\}^l \to \{0,1\}$ consists of the following PPT algorithms:

$\mathsf{Setup}(1^\lambda) \to (pp, msk)$. On input the security parameter 1^λ, the setup algorithm outputs the public parameters pp along with a master secret key msk.

$\mathsf{KeyGen}(msk, x) \to sk_x$**.** On input a master secret key msk and an attribute string $x \in \{0,1\}^l$, the key generation algorithm outputs a secret key sk_x.

$\mathsf{Enc}(f, \mu) \to ct_f$. On input a policy $f \in \mathcal{F}$ and a plaintext $\mu \in \{0,1\}$, the encryption algorithm outputs a ciphertext ct_f associated with f.

$\mathsf{Dec}(sk_x, ct_f) \to \mu/\perp$. On input a secret key sk_x and a ciphertext ct_f, the decryption algorithm outputs a bit $\mu \in \{0,1\}$ if $f(x) = 1$, else outputs the error symbol $\perp$.

$\mathsf{ReKeyGen}(sk_x, g) \to rk_{x\to g}$. Given a secret key sk_x and a policy $g \in \mathcal{F}$, this algorithm outputs a re-encryption key $rk_{x\to g}$.

$\mathsf{ReEnc}(rk_{x\to g}, ct_f) \to ct_g/\perp$. Given a re-encryption key $rk_{x\to g}$ and an original ciphertext ct_f, this algorithm outputs a new ciphertext ct_g associated with policy g if $f(x) = 1$, else outputs an error symbol $\perp$.

Correctness. A unidirectional, single-hop AB-PRE is correct if:

– For all $x \in \{0,1\}^l$ and $f \in \mathcal{F}$ for which $f(x) = 1$, and all $\mu \in \{0,1\}$, it holds that

$$\Pr[\mathsf{Dec}(sk_x, ct_f) \neq \mu] = \mathsf{negl}(\lambda),$$

where $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$ and $ct_f \leftarrow \mathsf{Enc}(f, \mu)$.

– For any $g \in \mathcal{F}$ and $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g)$, it holds that

$$\Pr[\mathsf{Dec}(sk_y, ct_g) \neq \mu] = \mathsf{negl}(\lambda),$$

if $g(y) = 1$ for $y \in \{0,1\}^l$, where $ct_g \leftarrow \mathsf{ReEnc}(rk_{x\to g}, ct_f)$.

Security Game for HRA. The adaptive HRA security game of a unidirectional, single-hop AB-PRE scheme between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ as below.

Phase 1 (Setup): This is the setup phase. The challenger generates (pp, msk) by running $\mathsf{Setup}(1^\lambda)$ algorithm and gives the public parameters pp to $\mathcal{A}$. Then, the challenger initializes a counter $\mathsf{numCt} := 0$, a policy-value store $\mathsf{C} := \emptyset$, a key list $\mathsf{K} := \emptyset$ and a set $\mathsf{Deriv} := \emptyset$.

Phase 2 (Oracle Query): This is the oracle query phase.

– $\mathcal{O}_{\mathsf{KeyGen}}(x)$: For a key query x, the challenger generates $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$ and adds (x, sk_x) in K. It gives sk_x to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{Enc}}(f, \mu)$: For an encryption query (f, μ), the challenger computes $ct_f \leftarrow \mathsf{Enc}(f, \mu)$, sets $\mathsf{numCt} := \mathsf{numCt} + 1$, adds ct_f in C with policy tuple (f, numCt), and gives (numCt, ct_f) to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{ReKey}}(x, g)$: For a re-encryption key query (x, g), we assume that there exists an attribute string y. If $y \notin \mathsf{K}$, the challenger records y such that it cannot be queried in $\mathcal{O}_{\mathsf{KeyGen}}$. There exist two cases:
  1) $g(y) = 0$, the challenger generates $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g)$.
  2) $g(y) = 1$, return $\perp$ if for any queried f in the store C such that i) $f(x) = 0$ and $x \in \mathsf{K} \wedge y \in \mathsf{K}$ or ii) $f(x) = 0$ and $x \notin \mathsf{K} \wedge y \in \mathsf{K}$, or iii) $f(x) = 1$ and $x \notin \mathsf{K} \wedge y \in \mathsf{K}$. Otherwise, the challenger produces $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g)$.

  After that, $\mathcal{C}$ gives $rk_{x\to g}$ to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{Cha}}(f^*, (\mu_0, \mu_1))$: This oracle can only be invoked once. For a challenge query $(f^*, (\mu_0, \mu_1))$, it requires $f^*(x) = 0$, where $x \in \mathsf{K}$. The challenger flips a bit $b \in \{0,1\}$, generates $ct_{f^*} \leftarrow \mathsf{Enc}(f^*, \mu_b)$, sets $\mathsf{numCt} := \mathsf{numCt} + 1$ and $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. It adds ct_{f^*} in C with policy tuple (f^*, numCt) and gives $(\mathsf{numCt}, ct_{f^*})$ to $\mathcal{A}$.

– $\mathcal{O}_{\mathsf{ReEnc}}((x, g), (f, k))$: For a re-encryption query $((x, g), (f, k))$, where $k \leq \mathsf{numCt}$. Suppose that there exist an attribute string y. If $y \notin \mathsf{K}$, the challenger records y such that it cannot be queried in $\mathcal{O}_{\mathsf{KeyGen}}$. Then the challenger does the following operations.
  1) If there is no value in C with policy tuple (f, k), return $\perp$.
  2) If $f(x) = 0$, return $\perp$.
  3) If $g(y) = 1 \wedge y \in \mathsf{K} \wedge k \in \mathsf{Deriv}$, return $\perp$.
  4) Otherwise, let ct_f be that value in the store C. The challenger produces $ct_g \leftarrow \mathsf{ReEnc}(rk_{x\to g}, ct_f)$ where $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g)$, sets $\mathsf{numCt} := \mathsf{numCt}+1$, adds ct_g in C with policy tuple (g, numCt). If $k \in \mathsf{Deriv}$, set $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. Finally, it gives (numCt, ct_g) to $\mathcal{A}$.

Phase 3 (Decision): This is the decision phase. $\mathcal{A}$ outputs a bit b' for b.

$\mathcal{A}$ wins the game if $b' = b$. We say that the AB-PRE is HRA-secure if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ winning in the game is negligible.