# Conditional Attribute-Based Proxy Re-Encryption: Definitions and Constructions from LWE

Lisha Yao[1], Jian Weng[2], Pengfei Wu[1], Xiaoguo Li[1], Yi Liu[2], Junzuo Lai[2], Guomin Yang[1], and Robert H. Deng[1]

[1] School of Computing and Information Systems, Singapore Management University, Singapore
{lishayao,pfwu,xiaoguoli,gmyang,robertdeng}@smu.edu.sg
[2] School of Cybersecurity, Jinan University, Guangzhou, China
cryptjweng@gmail.com, liuyi@jnu.edu.cn, laijunzuo@gmail.com

**Abstract.** Attribute-based proxy re-encryption (AB-PRE) is one of the essential variants for proxy re-encryption. It allows a proxy with a re-encryption key to transform a ciphertext associated with an access policy and decryptable by a *delegator* into another ciphertext associated with a new access policy, thereafter other *delegatees* can decrypt. However, with AB-PRE, the proxy is to switch the underlying policies of all ciphertexts indiscriminately. The delegator cannot decide which ciphertext would be transformed, taking no flexibility in controlling it for real use.

In this paper, we propose a notion of Conditional AB-PRE (CAB-PRE), supporting completely fine-grained control for ciphertexts, in both decryption and delegation. In CAB-PRE, the proxy can convert the underlying policy of a ciphertext only if this ciphertext satisfies a specific condition set by the delegator in the re-encryption key. We formalize the security of this notion in the honest re-encryption attacks (HRA) setting, and present a concrete construction secure under adaptive corruptions in the standard model. As a building block, we design an adaptively HRA-secure (ciphertext-policy) AB-PRE based on the learning with errors (LWE) problem, which solves an open problem left by Susilo *et al.* in ESORICS '21. Finally, we introduce a well-matched conditional delegation tailored to inner-product predicates and integrate it into this AB-PRE to derive our HRA-secure CAB-PRE scheme.

**Keywords:** Conditional AB-PRE · Honest re-encryption attacks · Learning with errors

## 1 Introduction

Proxy Re-Encryption (PRE) is a cryptographic primitive that enables the transfer of decryption rights between parties with the help of a proxy in a secure and privacy-preserving manner [4]. Specifically, a PRE scheme provides a proxy holding a re-encryption key with the ability to transform a ciphertext encrypted under a *delegator*'s key into the one under a *delegatee*'s key without revealing any information of the original message to the proxy. It is widely used in scenarios such as encrypted e-mail forwarding [30], online social networks [18], digital forensics evidence management [31], and CloudIoT platform [35].

One of the most prominent variants of PRE is Attribute-Based PRE (AB-PRE) [12,21,24,27,36], which can be derived from PRE and attribute-based encryption (ABE). This variant provides a more flexible access control than traditional PRE by allowing a proxy to convert the underlying policy of an ABE ciphertext. That is, a delegator's ciphertext encrypted under an access policy can be transformed into a delegatee's counterpart encrypted under a new access policy. A motivating scenario for AB-PRE applications is data sharing on the cloud. Suppose that a company's financial manager, Alice, is to travel for a while. Her incoming files are stored on the cloud and encrypted under distinct access policies $f_1, f_2, ..., f_s$. Alice has decryption rights for these files and wants to delegate her assistants to process files while she leaves. With an AB-PRE system, Alice can give the cloud server a re-encryption key involving a new access policy $g$, thereby enabling her assistants to decrypt the transformed files. During the transformation process, the cloud server can convert all of Alice's ciphertexts under $f_1, f_2, ..., f_s$ to ones associated with the new policy $g$ by re-encryption key without any other restrictions. In other words, as long as the ABE ciphertexts can be decrypted by Alice, her assistants with attributes satisfying $g$ can read them. Furthermore, if Alice just wants her assistants to handle a certain type of files, AB-PRE is obviously unable to fulfill such a requirement. Although AB-PRE supports secure data sharing, it allows the proxy to indiscriminately convert ciphertexts between different access policies and lacks fine-grained control in the ciphertext transformation.

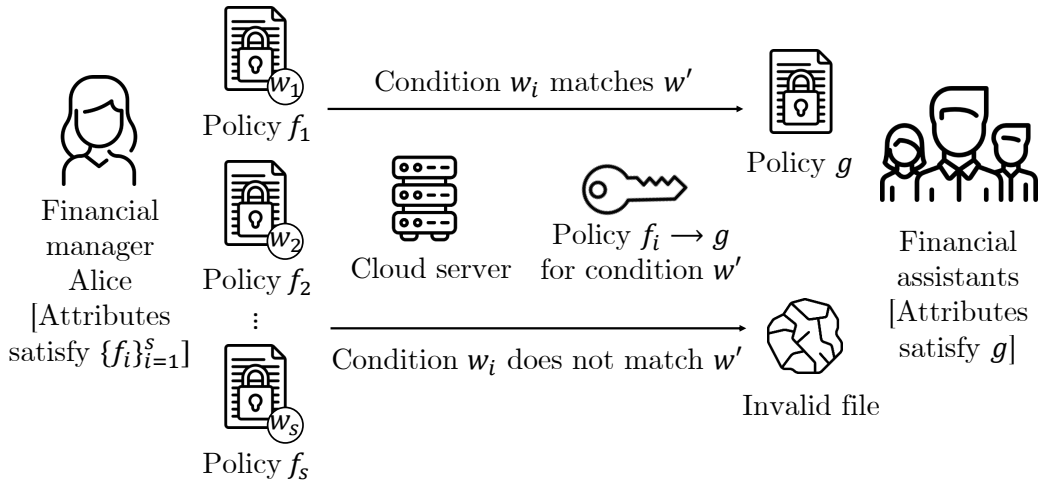


Fig. 1: Illustration of example for CAB-PRE.

### 1.1 Our Contributions

In this paper, we present a new notion called Conditional AB-PRE (CAB-PRE), with which Alice has completely fine-grained control over ciphertexts, in both decryption and delegation. Compared with the conventional AB-PRE schemes, we incorporate a conditional requirement, from the spirit of Conditional PRE (C-PRE) [20,39], into the ciphertext and re-encryption key. Taking the data sharing example above once again (see Fig. 1), with CAB-PRE, financial manager Alice's incoming files $ct_f^w$ are not only encrypted under different access policies $f \in \{f_1, f_2, ..., f_s\}$, but also attach some conditions $w \in \{w_1, w_2, ..., w_s\}$. These policies and conditions describe the characteristics of the decryptor and the file, respectively. For simplicity, we consider keywords as conditions. Assume that Alice's incoming files are $\{ct_{f_1}^{w_1}, ct_{f_2}^{w_2}, ct_{f_3}^{w_3}\}$, encrypted under policies "$f_1 = \texttt{financial} \wedge (\texttt{supervisor} \vee \texttt{manager})$", "$f_2 = \texttt{financial} \wedge \texttt{manager}$", "$f_3 = (\texttt{financial} \vee \texttt{product}) \wedge (\texttt{supervisor} \vee \texttt{manager})$" and keywords "$w_1 = w_3 = \texttt{Urgent}$", "$w_2 = \texttt{General}$". By passing to the cloud server a re-encryption key associated with a policy "$g = (\texttt{financial} \wedge \texttt{assistant}) \wedge (\texttt{Bob} \vee \texttt{Carol})$" and a keyword "$\texttt{Urgent}$", the cloud server can only convert files $ct_{f_1}^{w_1}$ and $ct_{f_3}^{w_3}$ into ones encrypted under the policy $g$ because these files contain the keyword "$\texttt{Urgent}$". Hence, the financial assistant Bob (or Carol) can read these two files, while the confidentiality of other files associated with the keyword "$\texttt{General}$" is kept.

As for the security of CAB-PRE, we formalize it in the HRA security model and propose the *first* adaptively secure construction based on the LWE problem [34]. Our approach is first to give an AB-PRE scheme and then construct a CAB-PRE scheme. It is worth noting that constructing an HRA-secure (ciphertext-policy) AB-PRE scheme is an open problem left by Susilo *et al.* [36]. At a high level, we integrate the idea of key switching into a (ciphertext-policy) ABE to produce a re-encryption key and sample a short vector to substitute the role of the secret key in the re-encryption key, thereby capturing such an AB-PRE scheme. We refer to the details in Section 1.2. Based on this AB-PRE, we obtain a CAB-PRE scheme by introducing an inner-product predicate to describe the delegation conditions with expressive descriptions and hiding properties. That is, the cloud server is unaware of any condition contained in the ciphertext during the ciphertext transformation process.

The underlying policy of our CAB-PRE scheme supports $t$-conjunctive normal form ($t$-CNF) predicates [37], and the conditional delegation supports inner-product predicates. These predicates imply bit-fixing and $t$-threshold policies, which have enough expressibility for real-world applications. The main contributions of this paper are summarized as follows.

1. We introduce a new primitive called CAB-PRE, in which an ABE ciphertext can only be transformed if the condition in the ciphertext matches the one in the re-encryption key. Then, we formalize its HRA security model, which has been demonstrated to be stronger than CPA security [9].
2. Designing as a building block, we construct the *first* adaptively HRA-secure AB-PRE scheme for $t$-CNF based on the LWE problem. Different from the known solution shown in Susilo *et al.* [36] in ESORICS '21, our scheme is designed on ciphertext policy and achieves adaptive security. Notably, it solves an open problem they left and thus is of independent interest.

Table 1: Correspondences between honest and corrupt users

| **Relation** | $y \in \mathsf{K} \wedge g(y) = 0$ | | $y \in \mathsf{K} \wedge g(y) = 1$ | | $y \notin \mathsf{K} \wedge g(y) = 0$ | | $y \notin \mathsf{K} \wedge g(y) = 1$ | |
|---|---|---|---|---|---|---|---|---|
| $x \in \mathsf{K} \wedge f(x) = 0$ | H $\to$ H | * | H $\to$ H | * | H $\to$ H | * | H $\to$ H | * |
| $x \in \mathsf{K} \wedge f(x) = 1$ | C $\to$ H | * | C $\to$ C | C $\to$ H | C $\to$ H | * | C $\to$ H | * |
| $x \notin \mathsf{K} \wedge f(x) = 0$ | H $\to$ H | * | H $\to$ H | * | H $\to$ H | * | H $\to$ H | * |
| $x \notin \mathsf{K} \wedge f(x) = 1$ | H $\to$ H | * | H $\to$ C | H $\to$ H | H $\to$ H | * | H $\to$ H | * |

$x \in \mathsf{K}$: an adversary possesses a user's secret key associated with an attribute $x$; $x \notin \mathsf{K}$: the adversary has no such key; $f(x) = 1$ (*resp.*, $g(y) = 1$): a user's secret key associated with an attribute $x$ (*resp.*, $y$) is capable of decrypting an original ciphertext (*resp.*, a transformed ciphertext) encrypted under an access policy $f$ (*resp.*, $g$); $f(x) = 0$ (*resp.*, $g(y) = 0$): the user's secret key cannot decrypt such a ciphertext. H: honest party, a user can be regarded honest, either if an adversary does not have the corresponding secret key or if this secret key cannot decrypt a ciphertext; C: corrupt party, a user can be regarded corrupt, if the adversary has the corresponding secret key and at the same time can decrypt a ciphertext; $i \to j$, where $i, j \in \{\mathrm{H}, \mathrm{C}\}$: an adversary initiates the queries from $i$ to $j$; $*$: a relation is the same as the gray area to the left.

3. Finally, we introduce a well-matched conditional delegation tailored to inner-product predicates, in which the predicate vector embedded in the ciphertext is hiding, and integrate it into our AB-PRE scheme to obtain our final CAB-PRE construction.

### 1.2 Technical Overview

Below we explain the technical challenges encountered in formalizing CAB-PRE and designing our construction, and give an overview of the way how to solve them.

**HRA Security Model for CAB-PRE.** The security notion of HRA is proposed by Cohen [9] and defined under selective corruptions. The main challenge in defining an HRA security model for CAB-PRE under adaptive corruptions is how to formalize honest and corrupt users in the system, and in further, define the oracles that the adversary queries. Below we highlight some key ideas of our solution.

*Formulate Honest and Corrupt Users.* In the PRE's security under selective corruptions, an adversary identifies a set of users that it wants to corrupt and captures their secret keys. Therefore, honest and corrupt users have been defined at the beginning of the security game. However, the security model of CAB-PRE cannot be formalized in selective corruptions, because the user's secret key is associated with multiple attributes, and a single attribute cannot uniquely identify a user. To implement the division of users, we introduce a key list $\mathsf{K}$ as an additional state to record a pair of an attribute and its corresponding secret key $(x, sk_x)$ that the adversary has queried.

We summarize all possible cases in Table 1 to define the different states of users and their relations. The gray columns indicate that a condition $w$ in the ciphertext satisfies a matching relation with a condition $w'$ in the re-encryption key, *i.e.*, $w \models w'$, and the white columns indicate that they do not satisfy, *i.e.*, $w \not\models w'$.

*Re-encryption Key and Re-encryption Oracles.* HRA security provides the adversary with a restricted re-encryption oracle which only re-encrypts honestly generated ciphertexts. Identical to CPA security, the adversary has no access to re-encryption key oracle for the users from honest to corrupt. However, our model enables the adversary to make re-encryption key queries under the case of $w \not\models w'$ at will.

For a re-encryption key query, the challenger would check three types of events, as shown in Table 1: (1) The status of the secret key possessed by the adversary for the original ciphertext, which depends on whether $x \in \mathsf{K}$ and $f(x) = 1$; (2) The status of the secret key possessed by the adversary for the transformed ciphertext, which depends on whether $y \in \mathsf{K}$ and $g(y) = 1$; (3) The transformable state of the original ciphertext, which depends on the matching relation between $w$ and $w'$. Combined with the above determinations, we can conclude that if $(x \notin \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1) \wedge (w \models w')$ is true, the challenger outputs an error symbol $\perp$; otherwise, it returns a re-encryption key generated by a regular way.

Similarly, we can define a re-encryption oracle. Compared with HRA security model for PRE introduced in Cohen [9], our new primitive involves access policy and delegation condition; thus, the re-encryption oracle terminates in two additional cases of $w \not\models w'$ and $f(x) = 0$.

**HRA-secure AB-PRE Construction.** We present an HRA-secure AB-PRE [3] based on LWE inspired by a (ciphertext-policy) ABE [37] and key switching technique [6]. Here, we recap this ABE scheme first. Basically, Tsabary utilizes a conforming cPRF for the predicate class $t$-CNF (denoted by $\mathcal{F}$ in this paper), and the policy circuits $U_{\sigma\to x}$, $U_{\sigma\to f}$, and $U_{f\to x}$. The public parameters of this scheme consist of two matrices $\mathbf{A}$ and $\mathbf{B}$, a vector $\mathbf{v}$, and the cPRF's public parameters *P.pp*. The master secret key comprises a trapdoor of matrix $\mathbf{B}$ and the cPRF's master secret key $\sigma$. For any policy $f \in \mathcal{F}$, one obtains an efficient computable matrix $\mathbf{A}_f$ using lattice evaluation. The ciphertext $(\mathbf{u}_0 \| \mathbf{u}_1, u_2)$ is a Dual-Regev encryption [14] with respect to a tuple $(\mathbf{B} \| [\mathbf{A}_f - s_f \otimes \mathbf{G}], \mathbf{v})$, where $s_f$ is generated by the cPRF's key simulation $\mathsf{P.KeySim}(P.pp, f)$, and $\mathbf{G}$ is a gadget matrix [28]. Similarly, one can capture another 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 preimage sampling $\mathsf{SamplePre}$ and satisfies $(\mathbf{B} \| \mathbf{A}_{x,r})\mathbf{k} = \mathbf{v}$, where $r$ is computed by the cPRF's evaluation $\mathsf{P.Eval}(\sigma, x)$. The decryption algorithm is the same as Dual-Regev decryption, namely computes $u_2 - (\mathbf{u}_0 \| \mathbf{u}'_1)\mathbf{k}$, in which $\mathbf{u}'_1$ is an output value of the circuits. The message can be successfully recovered iff $f(x) = 1 \wedge r \neq r'$, where $r' \leftarrow U_{f\to x}(s_f)$.

*First Attempt.* We adopt the idea of key switching technique in [6] to produce a re-encryption key

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

where $(\mathbf{H}, \mathbf{v}) = (\mathbf{B} \| [\mathbf{A}_g - s_g \otimes \mathbf{G}], \mathbf{v})$ is an encryption tuple related to a policy $g \in \mathcal{F}$, and $\mathbf{R}_1, \mathbf{R}_2, \mathbf{r}_3$ denote error terms. Then, a re-encrypted ciphertext is computed as $(\mathsf{BD}(\mathbf{u}_0 \| \mathbf{u}'_1) \| u_2) \cdot \mathbf{Z}$. The vector decomposition functions $\mathsf{P2}(\cdot)$ and $\mathsf{BD}(\cdot)$ are some subroutines of key switching and are used to prevent noise explosions in our construction.

Notably, the above construction can be proven to satisfy CPA security. However, it fails to provide HRA security. That is because the secret key $\mathbf{k}$ is embedded in re-encryption keys, the challenger is unable to answer the re-encryption queries from honest user to corrupt user, in the absence of secret keys.

*Second Attempt.* To address this issue, we make some changes that for $\mathbf{Z}$. Concretely, we choose a small value $\delta$ from a discrete Gaussian distribution and sample a short vector $\mathbf{d}$ by invoking $\mathsf{SamplePre}$ such that $(\mathbf{B} \| \mathbf{A}_{x,r})\mathbf{d} = \mathbf{v} \cdot \delta$, in which $\mathbf{d}$ is not a derivation of $\mathbf{k}$ [4]. Then, we generate a modified re-encryption key by replacing the second column $(\mathbf{R}_1\mathbf{v} + \mathbf{r}_3 - \mathsf{P2}(\mathbf{k}); 1)$ with $(\mathbf{R}_1\mathbf{v} + \mathbf{r}_3 - \mathsf{P2}(\mathbf{d}); \delta)$ in Equation 1.

**HRA-secure CAB-PRE Construction.** To incorporate a fine-grained condition on AB-PRE scheme, we find that the introduced condition needs to satisfy two requirements: (1) The condition embedded in the ciphertext should be compatible with other ciphertexts; (2) The embedding of the condition cannot affect the correctness of the re-encrypted ciphertext.

To achieve these requirements, we add an additional ciphertext component, namely $\mathbf{u}_3$, and design it as an LWE sample. Inspired by a cPRF for inner-product predicates [10], we embed a predicate vector into $\mathbf{u}_3$, because the condition in the ciphertext is eliminated when the inner product of two vectors is zero. Given $\mathbf{s}, \mathbf{D}, \mathbf{h}$ are chosen at random, $\boldsymbol{\alpha}$ is a predicate vector, and $\mathbf{e}_3$ is an error vector, $\mathbf{u}_3$ is computed as $\mathbf{s}^T(\mathbf{D} + \mathbf{h} \otimes \mathsf{P2}(\boldsymbol{\alpha})^T) + \mathbf{e}_3^T$. Accordingly, we modify the structure of the re-encryption key. Let $C_{\boldsymbol{\beta}}$ be an inner product predicate, we insert $\boldsymbol{\beta}$ into the re-encryption key

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

[3] The syntax and HRA security model of AB-PRE are provided in the Appendix A.

[4] Our proposed scheme relies on a crucial requirement that the norm of the vectors $\mathbf{k}$ and $\mathbf{d}$ should be bounded by $\tau\sqrt{m'+m}$ since they are generated from a Gaussian sampling with parameter $\tau$. 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 $\|\mathbf{d}\|_\infty \leq \tau\sqrt{m'+m} \cdot B$. This inequality $\tau\sqrt{m'+m} < \|\mathbf{d}\|_\infty \leq \tau\sqrt{m'+m} \cdot B$ holds with a non-negligible probability, where $\tau \in O(\lambda, n, k, (2m)^{d+3})$ and $B = B(n)$.

Table 2: Comparisons of properties between related AB-PRE schemes and ours

| Schemes | Type | Policy | Security | Assumption |
|---|---|---|---|---|
| Liang *et al.* [24] | CP | AND-gate | sCPA | ADBDH |
| Liang *et al.* [22] | CP | LSSS | aCCA | GSD, TPDH, $q$-PBDHE |
| Ge *et al.* [12] | KP | LSSS | aCCA | SD |
| Li *et al.* [21] | CP | AND-gate | sCPA | LWE |
| Luo *et al.* [27] | KP | Boolean circuit | sCPA | LWE |
| Susilo *et al.* [36] | KP | Boolean circuit | sHRA | LWE |
| Ours | CP | $t$-CNF | aHRA | LWE |

CP: ciphertext policy; KP: key policy; AND-gate: an access structure consisting of "AND" gates; LSSS: an access structure can be represented by a linear secret-sharing scheme; sCPA and aCCA: security notions, *i.e.*, selective security under chosen-plaintext attacks and adaptive security under chosen-ciphertext attacks. HRA security is stronger than CPA and weaker than CCA, and we refer interested readers to [9] for more details about their differences. ADBDH, GSD, TPDH, $q$-PBDHE and SD: classical number-theoretic assumptions, *i.e.*, augment decisional bilinear Diffie-Hellman, general subgroup decision, three party Diffie–Hellman, $q$-parallel bilinear Diffie-Hellman exponent, and subgroup decision.

where $(\mathbf{H}, \mathbf{V})$ is a Dual-Regev encryption tuple, $\mathbf{d}$ is sampled from a discrete Gaussian distribution *s.t.*, $(\mathbf{B}\|\mathbf{A}_{x,r})\mathbf{d} = (\mathbf{V} + \mathbf{D}) \cdot \mathsf{BD}(\boldsymbol{\beta})$, and $\boldsymbol{\gamma} \xleftarrow{\$} \{0,1\}^{l\lceil \log q \rceil}$. We may note that only if $\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle = 0$, the ciphertexts can be re-encrypted. Since the ciphertext components used for decryption do not change before or after re-encryption, except that their error boundaries are different, our scheme maintains only one decryption algorithm.

### 1.3 Related Work

Many efforts have been made to AB-PRE and C-PRE based on classical number-theoretic and post-quantum assumptions. Let us recall existing works on them.

**AB-PRE**. Liang *et al.* [24] were the first to introduce the primitive of AB-PRE and realize fine-grained decryptions of the users. They proposed a concrete construction based on a bilinear map and proved it is CPA secure in the standard model. Following their work, Liang *et al.* [22] and Ge *et al.* [12] strengthen their scheme to be CCA secure, built in the composite order bilinear group. In addition, many AB-PRE schemes with various functionalities are proposed for practical application scenarios, such as collusion resistance [32], verifiable [11], and revocable [13] AB-PRE schemes. To resist the attack of quantum computers, Li *et al.* [21] presented a (ciphertext-policy) AB-PRE scheme under the LWE problem, which supports AND-gates on positive and negative attributes. Based on the ABE scheme introduced by Boneh *et al.* [5], Luo *et al.* [27] were the first to propose multi-hop (key-policy) AB-PRE scheme based on LWE, which is proven to be CPA secure. Subsequently, Susilo *et al.* [36] presented an HRA-secure (key-policy) AB-PRE scheme under selective corruptions and left an open problem to construct an HRA-secure (ciphertext-policy) AB-PRE, which is addressed in our work.

**C-PRE**. Weng *et al.* [39, 40] proposed the notion of C-PRE and empowered users with fine-grained delegations. They gave a CCA-secure construction using bilinear pairings in the random oracle model. To extend the expressiveness of the condition, many variants of C-PRE are proposed. Liang *et al.* [23] introduced the concept of identity-based C-PRE and presented a CCA-secure scheme in the standard model. Yang *et al.* [41] formalized the notion of (ciphertext-policy) attribute-based C-PRE and proposed a concrete scheme that supports an access tree and satisfies CPA security in the generic group model. In addition to that, there are conditional proxy broadcast re-encryption [26], universal C-PRE [17], and threshold-based C-PRE [16]. Aiming at a hierarchical user management system, Wang *et al.* [38] presented a lattice-based hierarchical identity-based C-PRE scheme by combining a hierarchical identity-based encryption and proxy re-encryption. After that, Li *et al.* [20] constructed the first lattice-based fuzzy C-PRE scheme that does not require the conditions in ciphertexts and re-

Table 3: Comparisons of storage costs between related AB-PRE schemes and ours

| **Param.** | **Li *et al.* [21]** | **Luo *et al.* [27]** | **Susilo *et al.* [36]** | **Ours** |
|---|---|---|---|---|
| PP | $O(2l \cdot n^2 \log^2 q)$ | $O((l+2) \cdot n^2 \log^2 q)$ | $O((l+3) \cdot n^2 \log^2 q)$ | $O((\lambda+1) \cdot n^2 \log^2 q)$ |
| MSK | $O(2l \cdot n^2 \log^3 q)$ | $O(n^2 \log^3 q)$ | $O(n^2 \log^3 q)$ | $O(n^2 \log^3 q)$ |
| SK | $O(l \cdot n \log^2 q)$ | $O(4n^2 \log^3 q)$ | $O(2n^2 \log^3 q)$ | $O(4n^2 \log^3 q)$ |
| CT | $O(2l \cdot n \log^2 q)$ | $O((l+2) \cdot n \log^2 q)$ | $O((l+2) \cdot n \log^2 q)$ | $O((l_f+1) \cdot n \log^2 q)$ |
| RK | $O(2l \cdot n^2 \log^3 q)$ | $O(4n^2 \log^3 q)$ | $O((l+2) \cdot n^2 \log^4 q)$ | $O((l_g+1) \cdot n^2 \log^4 q)$ |
| CT' | $O(2l \cdot n \log^2 q)$ | $O(3n \log^2 q)$ | $O((l+2) \cdot n \log^2 q)$ | $O((l_f+1) \cdot n \log^2 q)$ |

$n$: the dimension of LWE samples; $l$: the length of attribute vectors; $\lambda$: security parameter; $l_f$ and $l_g$: the length of constrained key in conforming cPRF, associated with functions $f$ and $g$, respectively.

encryption keys to match precisely. Liang *et al.* [25] proposed two LWE-based (key-policy) attribute-based C-PRE schemes consisting of single-hop and multi-hop by applying key switching.

To have a better understanding of our scheme, we compare our proposed AB-PRE scheme with counterparts in terms of properties and storage costs in Table 2 and Table 3, respectively. Table 2 shows that the predicate expressivity supported by our scheme is weaker than Liang *et al.* [22], Ge *et al.* [12], Luo *et al.* [27] and Susilo *et al.* [36]. Nevertheless, our construction achieves a higher security level than Luo *et al.* [27], and Susilo *et al.* [36] and has the feature of quantum-resistance compared with Liang *et al.* [22] and Ge *et al.* [12]. Table 3 gives a storage comparison with three LWE-based AB-PRE schemes (*i.e.*, Li *et al.* [21], Ge *et al.* [27] and Susilo *et al.* [36]), regarding the size of public parameters (PP), mater secret key (MSK), secret key (SK), original ciphertext (CT), re-encryption key (RK) and re-encrypted ciphertext (CT'). The results show that our scheme maintains the same order of magnitude in parameter, key, and ciphertext sizes as these schemes without incurring more computational overhead.

### 1.4 Organization

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

## 2 Preliminaries

In this section, we introduce some cryptography knowledge and lattice background 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 (cPRF [10,37]).** *Let $\mathcal{F}$ be a family of functions with domain $\{0,1\}^l$ and range $\{0,1\}$. A 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 $1^\lambda$, 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* $\Pi_{\mathsf{cPRF}}$ *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* $b \xleftarrow{\$} \{0,1\}$. *If* $b = 1$, *it evaluates* $y^* \leftarrow \mathsf{Eval}(msk, x^*)$. *Otherwise, it samples* $y^* \xleftarrow{\$} \{0,1\}^k$. *Finally,* $\mathcal{C}$ *returns* $y^*$ *to* $\mathcal{A}$.
- **Phase 2** : $\mathcal{A}$ *continues to make queries as the 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 [37]).** *We call a cPRF scheme 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* $b \xleftarrow{\$} \{0,1\}$ *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 [10,37]).** *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 [37] 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 construction of bit-fixing cPRF for a constant number of keys [10].

### 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 $(\cdot \parallel \cdot)$ (*resp.*, $(\cdot\ ;\ \cdot)$) denote the horizontal concatenation (*resp.*, vertical concatenation) of vectors or matrices. For a distribution on set $X$, we use $x \xleftarrow{\$} X$ 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 a matching relation (*resp.*, mismatching relation) between two conditions. We use $\overset{c}{\approx}$ to indicate that the two distributions are 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: $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\frac{\|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2})$ 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 $\sigma$ 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 ( [1,29]).** *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 ( [7,37]).** *A distribution $\chi$ supported over $\mathbb{Z}$ is $(B,\epsilon)$-bounded, if we have* $\Pr[x \xleftarrow{\$} \chi : |x| > B] < \epsilon$.

**Definition 6 ( [7, 37]).** *A distribution $\tilde{\chi}$ supported over $\mathbb{Z}$ is $(B,\epsilon)$-swallowing if for all $y \in [-B, B] \cap \mathbb{Z}$, we have that $\tilde{\chi}$ and $y + \tilde{\chi}$ are within $\epsilon$ 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 [28]).** *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 $\tau$, 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 $\mathbf{T'_A}$ 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 $\|\widetilde{\mathbf{T'_A}}\| < \tau\sqrt{m}$ with all but negligible probability.*

We introduce the abstraction form of lattice evaluation as extracted in [37].

**Theorem 1 (Lattice Evaluation [37]).** *Let $n, q, l, k \in \mathbb{N}$ and $m = n\lceil \log q \rceil$, there exist two deterministic algorithms called* $\mathsf{EvalF}$ *and* $\mathsf{EvalFX}$*, 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 $\mathbf{H} \leftarrow \mathsf{EvalF}(f, \mathbf{A})$ and $\hat{\mathbf{H}} \leftarrow \mathsf{EvalFX}(f, x, \mathbf{A})$ are both in $\mathbb{Z}^{ml \times mk}$ and it holds that $\|\mathbf{H}\|_\infty$, $\|\hat{\mathbf{H}}\|_\infty \leqslant (2m)^d$ and*

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

*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 $\mathbf{H}_f \leftarrow \mathsf{EvalF}(f, \mathbf{A})$, $\mathbf{H}_g \leftarrow \mathsf{EvalF}(g, \mathbf{AH}_f)$ and $\mathbf{H}_{g \circ f} \leftarrow \mathsf{EvalF}(g \circ f, \mathbf{A})$ satisfy $\mathbf{H}_f\mathbf{H}_g = \mathbf{H}_{g \circ f}$.*

**Learning With Errors.** For the learning with errors (LWE) problem [34], we will use its decisional version, Hermite normal form (HNF) and lossy mode, in this work.

**Definition 7 (Decisional LWE (DLWE) and Its HNF [2,34]).** *Let $\lambda$ 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 $(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e})$ and $(\mathbf{A}, \mathbf{u})$ are computationally indistinguishable. The form of HNF-LWE is identical to the above except for $\mathbf{s} \xleftarrow{\$} \chi^n$.*

**Corollary 1 ( [33, 34, 37]).** *For all $\epsilon > 0$, there exist functions $q = q(n) \leq 2^n$, $\chi = \chi(n)$ such that $\chi$ 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 $\mathsf{GapSVP}_\gamma$ and the quantum hardness of $\mathsf{SIVP}_\gamma$ for $\gamma = 2^{\Omega(n^\epsilon)}$.*

**Definition 8 (Lossy Mode for LWE [19]).** *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 [6]).** *There are two deterministic functions defined as below:*

- $\mathsf{BD}(\mathbf{v})$ : *Given a vector $\mathbf{v} \in \mathbb{Z}_q^n$, let $v_i^T \in \{0,1\}^n$ [5] 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 $\mathsf{BD}(\mathbf{v}) = (\mathsf{BD}(\mathbf{v}^T))^T$.*
- $\mathsf{P2}(\mathbf{x})$ : *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 $\mathsf{BD}(\mathbf{v}) \cdot \mathsf{P2}(\mathbf{x}) = \mathbf{v} \cdot \mathbf{x} \bmod q$.*

**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

In this paper, we restrict our attention to unidirectional, single-hop PRE [6]. We subsequently formally define the syntax of unidirectional, single-hop CAB-PRE for ciphertext policy and its HRA security model under adaptive corruptions.

### 3.1 Conditional Attribute-Based Proxy Re-encryption

In this section, we propose a new primitive called CAB-PRE by extending the notion of AB-PRE to one with fine-grained delegations. The single-hop PRE scheme usually implies that the ciphertext structure is different before and after transformation, and it can be achieved by furnishing two levels of encryption algorithms. In our syntax of CAB-PRE, the first-level ciphertext is generated by an access policy and a message, and it cannot be re-encrypted anymore. The second-level ciphertext contains some specified conditions and can be re-encrypted to the first-level ciphertext only if a re-encryption key corresponds to the same conditions. Accordingly, we define two decryption algorithms for distinct ciphertexts.

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

[6] 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.

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

$\mathsf{Setup}(1^\lambda) \to (pp, msk)$. *On input a security parameter $1^\lambda$, 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 message $\mu \in \{0,1\}$, the first-level encryption algorithm outputs a first-level ciphertext $ct_f^1$ associated with the policy $f$, which cannot be re-encrypted anymore.*

$\mathsf{Enc}_2(f, w, \mu) \to ct_f^2$. *On input a policy $f \in \mathcal{F}$, a condition $w$ and a message $\mu \in \{0,1\}$, the second-level encryption algorithm outputs a second-level ciphertext $ct_f^2$ associated with the policy $f$ and condition $w$, which can be further re-encryptd to a first-level ciphertext.*

$\mathsf{Dec}_1(sk_x, ct_f^1) \to \mu/\perp$. *On input a secret key $sk_x$ and a first-level ciphertext $ct_f^1$, the first-level decryption algorithm outputs a bit $\mu \in \{0,1\}$ if $f(x) = 1$ [7], else outputs an error symbol $\perp$.*

$\mathsf{Dec}_2(sk_x, ct_f^2) \to \mu/\perp$. *On input a secret key $sk_x$ and a second-level ciphertext $ct_f^2$, the second-level decryption algorithm outputs a bit $\mu \in \{0,1\}$ if $f(x) = 1$, 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 the 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:*

- *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*
$$\Pr[\mathsf{Dec}_1(sk_x, ct_f^1) \neq \mu] = \mathsf{negl}(\lambda), \Pr[\mathsf{Dec}_2(sk_x, ct_f^2) \neq \mu] = \mathsf{negl}(\lambda),$$
*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 all $g \in \mathcal{F}$, and any condition $w'$ and $rk_{x\to g}^{w'} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$, it holds that*
$$\Pr[\mathsf{Dec}_1(sk_y, ct_g^1) \neq \mu] = \mathsf{negl}(\lambda),$$
*if $g(y) = 1$ and $w \models w'$ for $y \in \{0,1\}^l$, where $ct_g^1 \leftarrow \mathsf{ReEnc}(rk_{x\to g}^{w'}, ct_f^2)$.*

### 3.2 Security Model

Now we formally give the HRA security model under the adaptive corruptions 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 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 $\mathsf{K}$. It gives $sk_x$ to $\mathcal{A}$.*
- $\mathcal{O}_{\mathsf{Enc}_2}(f, w, \mu)$ : *For a second-level encryption query $(f, w, \mu)$, the challenger computes $ct_f^2 \leftarrow \mathsf{Enc}_2(f, w, \mu)$, sets $\mathsf{numCt} := \mathsf{numCt} + 1$, adds $ct_f^2$ in $\mathsf{C}$ with policy tuple $(f, w, \mathsf{numCt})$, and gives $(\mathsf{numCt}, ct_f^2)$ to $\mathcal{A}$.*

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

- $\mathcal{O}_{\mathsf{ReKey}}(x, g, w')$ : *For a re-encryption key query* $(x, g, w')$*, if exist* $y \in \mathsf{K}$ *and* $(f, w, \cdot) \in \mathsf{C}$ *s.t.,* $(x \notin \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1) \wedge (w \models w')$ *holds, then the challenger returns* $\perp$*; otherwise, it generates* $rk^{w'}_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$ *and gives it 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* $ct^2_{f^*} \leftarrow \mathsf{Enc}_2(f^*, w^*, \mu_b)$*, sets* $\mathsf{numCt} := \mathsf{numCt} + 1$ *and* $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$*. It adds* $ct^2_{f^*}$ *in* $\mathsf{C}$ *with policy tuple* $(f^*, w^*, \mathsf{numCt})$ *and gives* $(\mathsf{numCt}, ct^2_{f^*})$ *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* $k \leq \mathsf{numCt}$*. The challenger does the following operations.*
  1) *If* $w \not\models w'$*, return* $\perp$*.*
  2) *If there is no value in* $\mathsf{C}$ *with policy tuple* $(f, w, k)$*, return* $\perp$*.*
  3) *If* $f(x) = 0$*, return* $\perp$*.*
  4) *If exists* $y \in \mathsf{K}$ *s.t.,* $(y \in \mathsf{K} \wedge g(y) = 1) \wedge (k \in \mathsf{Deriv})$ *holds, return* $\perp$*.*
  5) *Otherwise, let* $ct^2_f$ *be that value in the store* $\mathsf{C}$*. The challenger produces* $ct^1_g \leftarrow \mathsf{ReEnc}(rk^{w'}_{x\to g}, ct^2_f)$*, where* $rk^{w'}_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g, w')$*, sets* $\mathsf{numCt} := \mathsf{numCt} + 1$*, adds* $ct^1_g$ *in* $\mathsf{C}$ *with policy tuple* $(g, \mathsf{Null}, \mathsf{numCt})$*. If* $k \in \mathsf{Deriv}$*, set* $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$*. Finally, it gives* $(\mathsf{numCt}, ct^1_g)$ *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* $1^\lambda$*, 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* $\mathsf{negl}(\lambda)$ *s.t.,*

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

## 4 Adaptively HRA-Secure AB-PRE from LWE

In this section, we present a construction of AB-PRE scheme for the family of $t$-CNF predicates as a building block, by integrating a (ciphertext-policy) ABE scheme [37] with key switching technique [6]. Then based on the LWE problem, we prove the HRA security under adaptive corruptions for our AB-PRE scheme. Notably, this scheme resolves the open problem introduced by Susilo *et al.* [36] 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.

### 4.1 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 length of $\mathsf{P}$'s master secret key is $\lambda$. 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 $\sigma = P.msk$. $n, q, m, m', \tau, \chi, \tilde{\chi}$ are parameters. Invoke $(\mathbf{B}, \mathbf{T_B}) \leftarrow \mathsf{TrapGen}(1^n, m', q)$. 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 $msk = (\sigma, \mathbf{T_B})$.

$\mathsf{KeyGen}(msk, x)$ : 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 $\mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}} \leftarrow \mathsf{RandBasis}(\mathbf{B}\|\mathbf{A}_{x,r}, \mathsf{ExtendRight}(\mathbf{B}, \mathbf{T_B}, \mathbf{A}_{x,r}), \tau)$ 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 $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, u_2)$.

$\mathsf{Dec}(sk_x, ct_f)$**:** 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 $\mathbf{A}_f$ the same as $\mathsf{KeyGen}$ and $\mathsf{Enc}$, 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 the transformed ciphertext involves a coefficient factor $\delta$, and the correct recovery of the message $\mu$ is contingent upon the appropriate tailoring of the error 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 $\mathsf{KeyGen}$. 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 $\mathsf{KeyGen}, \mathsf{Enc}$ and $\mathsf{Dec}$. 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 $ct_g = (s_g, \delta, ct_{f\to g})$.

### 4.2 Correctness and Choice of Parameters

**Theorem 2.** *The AB-PRE scheme is correct with respect to $f \in \mathcal{F}$ under proper parameters as below.*

*Proof.* First, we direct our focus towards the correctness of the original ciphertext. For the security parameter $\lambda$, 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 [37], the probability of $r = r'$ is almost negligible, as evidenced by a reduction to the pseudorandomness game of the conforming cPRF $\mathsf{P}$. 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 $\mathsf{P}$ (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 $d_{\mathsf{Con}}$ and $d_{\mathsf{ConEval}}$, respectively. These depths are limited by the depth $d = \mathsf{poly}(\lambda)$ of $U_{\sigma \to x}$, as $\mathsf{P}$ 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 $\mathbf{e}_1 \in \tilde{\chi}^{m l_f}, \hat{\mathbf{H}}_{s_f \to r'} \in \mathbb{Z}^{m l_f \times mk}, \hat{\mathbf{H}}_{r,r'} \in \mathbb{Z}^{mk \times m}$ 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

- $q, \chi, B$ 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.

Subsequently, we show the correctness of the re-encrypted ciphertext. Given an original ciphertext associated with a function $f$, it can be efficiently re-encrypted to another ciphertext associated with a 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 $|\bar{e}_2| \leq B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}$.

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

$$u = c_2 - (\mathbf{c}_0 \| \mathbf{c}_1 \hat{\mathbf{H}}_{s_g \to \bar{r}'} \hat{\mathbf{H}}_{\bar{r},\bar{r}'})\bar{\mathbf{k}} = \mu \lfloor q/2 \rceil \cdot \delta + \bar{e}_2 - (\bar{\mathbf{e}}_0^T \| \bar{\mathbf{e}}_1')\bar{\mathbf{k}}.$$

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

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

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. □

### 4.3 Security Proof

For now, we will construct two efficient randomized algorithms $\mathsf{KeyGen}_1$ and $\mathsf{KeyGen}_2$ 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 $\mathsf{KeyGen}$, and captures $\hat{sk}_x$ by calling $\mathsf{SampleLeft}(\mathbf{B}, \mathbf{T}_\mathbf{B}, \mathbf{A}_{x,r}, \mathbf{v}', \tau)$, 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 $\mathsf{KeyGen}$, 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 $\mathbf{v}'$.

*Remark 1.* 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 $\mathsf{KeyGen}_1$ or $\mathsf{KeyGen}_2$ are statistically indistinguishable. In fact, $\hat{sk}_x$ can be regarded a substitute for real secret key and is used to generate a re-encryption key. However, it cannot used to decryption.

**Theorem 3.** *For a class of functions $\mathcal{F}$, if $\mathsf{P}$ 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 given in Appendix A. 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.

$\mathbf{Game}_1$ : In this game, we change the way the matrix $\mathbf{Z}$ is created, except in the case of $(x \in \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1)$. The challenger chooses $\mathbf{Z}_1 \xleftarrow{\$} \chi^{(m'+m)\lceil \log q \rceil \times (m'+ml_g)}$, $\mathbf{z}_2 \xleftarrow{\$} \chi^{(m'+m)\lceil \log q \rceil}$ and $\delta \xleftarrow{\$} \chi$, generates a component of re-encryption key as

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

Recall this component in the real scheme is

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

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 [25]). Therefore, we have $|\Pr[S_0] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$.

**Game**$_2$ : In this game, we change the way the matrix $\mathbf{Z}$ is generated, in the case of $(x \in \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1)$. When $\mathcal{A}$ makes a query on $(x, g)$, the challenger invokes $\mathsf{KeyGen}_1(msk, x)$ algorithm (here $msk = (\sigma, \mathbf{T_B})$) to obtain $\hat{sk}_x (= \mathbf{d})$, and then computes $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(\hat{sk}_x, g)$. Recall in the previous game, $\mathbf{d}$ is sampled by $\mathsf{ExtendRight}$, $\mathsf{RandBasis}$ and $\mathsf{SamplePre}$ algorithms. According to Lemma 2,Lemma 5 and Lemma 7, the distribution of $\mathbf{d}$ is $2^{-\Omega(n)}$-close $\mathcal{D}_{\Lambda_q^{\mathbf{y}'}(\mathbf{B}\|\mathbf{A}_{x,r}),\tau}$. Therefore, we have $|\Pr[S_1] - \Pr[S_2]| \leq \mathsf{negl}(\lambda)$.

**Game**$_3$ : 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 $\mathsf{P}$ and this is negligible. Suppose there exists an adversary $\mathcal{A}_0$ such that $|\Pr[S_2] - \Pr[S_3]|$ is non-negligible, we can build an algorithm $\mathcal{C}$ that wins the key simulation game of $\mathsf{P}$ 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 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**$_2$ or **Game**$_3$ 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**$_2$ to $\mathcal{A}_0$. Otherwise, $\mathcal{C}$ provides a view of **Game**$_3$ 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 $|\Pr[S_2] - \Pr[S_3]|$ is non-negligible, then $\mathcal{C}$ could break the key simulation game with a non-negligible advantage.

**Game**$_4$ : 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 $|\Pr[S_3] - \Pr[S_4]| \leq \mathsf{negl}(\lambda)$.

**Game**$_5$ : 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

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

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 $d_{\mathsf{Con}}$ 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 $|\Pr[S_4] - \Pr[S_5]| \leq \mathsf{negl}(\lambda)$.

**Game**$_6$ : In this game, we change the way key queries are answered. When $\mathcal{A}$ queries on $x$, the challenger evaluates $r \leftarrow \mathsf{P.Eval}(\sigma, x)$, $\hat{\mathbf{H}}_{msk\to r} \leftarrow \mathsf{EvalFX}(U_{\sigma\to x}, \sigma, \mathbf{A})$ and $\hat{\mathbf{H}}_{r,r} \leftarrow \mathsf{EvalFX}(I_r, r, \mathbf{A}_x)$, and computes

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

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 $\mathbf{B}\mathbf{R}\hat{\mathbf{H}}_{msk\to r}\hat{\mathbf{H}}_{r,r} = \mathbf{A}_{x,r} - \mathbf{G}$ due to $\mathbf{A} - \sigma \otimes \mathbf{G} = \mathbf{B}\mathbf{R}$. Note that

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

and

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

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

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

The responses to key queries are statistically close to those in the previous game, the adversary's advantage is at most negligibly different from its advantage in **Game**$_5$. Therefore, we have $|\Pr[S_5] - \Pr[S_6]| \le \mathsf{negl}(\lambda)$.

**Game**$_7$ : In this game, we change again the way of generating $\mathbf{Z}$ under the case of $(x \in \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1)$. The challenger runs $\mathsf{KeyGen}_2(msk, x)$ algorithm to compute $\hat{sk}_x (= \mathbf{d})$, thereby captures $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(\hat{sk}_x, g)$. Therefore, we have $|\Pr[S_6] - \Pr[S_7]| \le \mathsf{negl}(\lambda)$.

**Game**$_8$ : 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. Therefore, we have $|\Pr[S_7] - \Pr[S_8]| \le \mathsf{negl}(\lambda)$.

**Game**$_9$ : 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 $|\Pr[S_8] - \Pr[S_9]|$ 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 **Game**$_8$ and **Game**$_9$. 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. It assembles public parameters $pp$: compute $(P.pp, P.msk) \leftarrow \mathsf{P.Setup}(1^\lambda)$, set $\sigma = P.msk$ and $\mathbf{A} = \mathbf{B}\mathbf{R} + \sigma \otimes \mathbf{G}$, where $\mathbf{R} \xleftarrow{\$} \{1, -1\}^{m'\times m\lambda}$. $\mathcal{B}$ gives $pp = (\mathbf{B}, \mathbf{A}, \mathbf{v}, P.pp)$ to $\mathcal{A}_1$. The master secret key contains only $\sigma$. 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 $\mathcal{A}_1$'s key queries, encryption queries, re-encryption key queries and re-encryption queries as in **Game**$_9$, except for challenge query.

- $\mathcal{O}_{\mathsf{Cha}}(f^*, (\mu_0, \mu_1))$: To generate a challenge ciphertext, $\mathcal{B}$ first picks a random bit $b \in \{0, 1\}$, computes $s_{f^*}, \mathbf{u}_1^*$ as in **Game**$_9$ 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 $\mathsf{C}$ with policy tuple $(f^*, \mathsf{numCt})$ 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 $\mathbf{Game}_8$ or $\mathbf{Game}_9$. $\mathcal{B}$ 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 $\mathbf{Game}_8$. Otherwise (*i.e.*, $(\mathbf{u}_0\|u_2') \xleftarrow{\$} \mathbb{Z}_q^{1\times(m'+1)}$), it corresponds to $\mathbf{Game}_9$. 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 $|\Pr[S_8] - \Pr[S_9]| \leq \mathsf{negl}(\lambda)$.

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

## 5 Adaptively HRA-Secure CAB-PRE from LWE

In this section, We 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.

### 5.1 Construction

Let $C_{\boldsymbol{\beta}} : \mathbb{Z}_q^l \to \{0,1\}$ be an inner-product predicate [8] with dimension $l$, which evaluates $C_{\boldsymbol{\beta}}(\boldsymbol{\alpha}) = 1$ when given an input $\boldsymbol{\alpha} \in \mathbb{Z}_q^l$ such that $\langle\boldsymbol{\alpha}, \boldsymbol{\beta}\rangle = 0$. 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 $msk = (\sigma, \mathbf{T}_{\mathbf{B}})$.

$\mathsf{KeyGen}(msk, x)$ : 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 $ct_f^1 = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2)$.

$\mathsf{Enc}_2(f, \boldsymbol{\alpha}, \mu)$**:** 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 $\mathsf{Enc}_1$ 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 $sk_x = (r, \mathbf{T}_{\mathbf{B}\|\mathbf{A}_{x,r}}, \mathbf{K})$, $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2)$ (if $ct_f$ is a first-level ciphertext) or $ct_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ (if $ct_f$ is a second-level ciphertext). 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 operation remains constant regardless of the first-level or second-level ciphertexts. Decrypting the original ciphertext is subject to an upper limit of $q/4$. However, when decrypting a transformed ciphertext, a coefficient factor linked to the predicate vector $\boldsymbol{\beta}$ is involved. In such cases, adjusting the error bound appropriately ensures the accurate retrieval of the message $\mu$.

$\mathsf{ReKeyGen}(sk_x, g, \boldsymbol{\beta})$**:** 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

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

[8] 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 error bound and ensure the decryption correct. Without loss of generality, we assume that of $\beta_l = q - 1$ in decryption phase.

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

$\mathsf{ReEnc}(rk^{\boldsymbol{\beta}}_{x\to g}, ct^2_f)$**:** Parse $rk^{\boldsymbol{\beta}}_{x\to g} = (s_g, r, \boldsymbol{\beta}, \gamma, \mathbf{Z})$ and $ct^2_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$. 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 $ct^1_g = (s_g, \boldsymbol{\beta}, \gamma, ct_{f\to g})$.

### 5.2 Correctness and Choice of Parameters

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

*Proof.* The original ciphertext contains either the first-level ciphertext or the second-level ciphertext, as they have same decryption operations. According to the parameters given in Section 4.2, the correctness is as follows.

First, for the security parameter $\lambda$, 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 $(pp, msk) \leftarrow \mathsf{Setup}(1^\lambda)$, $sk_x \leftarrow \mathsf{KeyGen}(msk, x)$, $ct^1_f \leftarrow \mathsf{Enc}_1(f, \mu)$ and $ct^2_f \leftarrow \mathsf{Enc}_2(f, \boldsymbol{\alpha}, \mu)$, then we have $\mu \leftarrow \mathsf{Dec}(sk_x, ct_f \in \{ct^1_f, ct^2_f\})$ 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

$$\mathbf{u}_2 - (\mathbf{u}_0\|\mathbf{u}_1\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'})\mathbf{K} = \mu(\mathbf{0}\|\mathbf{g}^T) + \mathbf{e}^T_2 - (\mathbf{e}^T_0\|\mathbf{e}'_1)\mathbf{K},$$

where $\mathbf{e}'_1 = \mathbf{e}^T_1\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'}$ 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 $l, m', l_f, k \in O(n, \lceil\log q\rceil)$, $\tilde{B} \in O(B, n)$ and $\tau \in O(\lambda, k, (2m)^{d+3})$, then

$$\begin{aligned}\|\mathbf{e}^T_2 - (\mathbf{e}^T_0\|\mathbf{e}'_1)\mathbf{K}\|_\infty &\le \|\mathbf{e}^T_2\|_\infty + (m'\|\mathbf{e}^T_0\|_\infty + m\|\mathbf{e}'_1\|_\infty)\|\mathbf{K}\|_\infty \\ &\le l\,\lceil\log q\rceil B + (m'B + m^3 l_f k\tilde{B}(2m)^{d_{\mathsf{ConEval}}+1})\tau\sqrt{(m'+m)\cdot l\,\lceil\log q\rceil} \\ &\le B\cdot\mathsf{poly}(n, \lceil\log q\rceil)\cdot(2m)^{d_{\mathsf{ConEval}}+d+4}.\end{aligned}$$

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

$$\|\mathbf{e}^T_2 - (\mathbf{e}^T_0\|\mathbf{e}'_1)\mathbf{k}\|_\infty \le B\cdot\mathsf{poly}(n\lceil\log q\rceil)\cdot(2m)^{d_{\mathsf{ConEval}}+d+4} < q/8,$$

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 a function $f$ can be efficiently re-encrypted to a first-level ciphertext associated with a function $g$, where the delegation condition is described as the inner-product predicates between two vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ over $\mathbb{Z}^l_q$. Specifically, parse the re-encryption key $rk^{\boldsymbol{\beta}}_{x\to g} = (s_g, r, \boldsymbol{\beta}, \gamma, \mathbf{Z})$ and the second-level ciphertext $ct^2_f = (s_f, \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$, 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

$$\mathbf{u}'_1 = \mathbf{u}_1\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'} = \mathbf{s}^T\mathbf{A}_{x,r} + \mathbf{e}^T_1\hat{\mathbf{H}}_{s_f\to r'}\hat{\mathbf{H}}_{r,r'}$$

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

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

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 $\|\bar{\mathbf{e}}_2\|_\infty \leq B \cdot \mathsf{poly}(n, \lceil \log q \rceil) \cdot (2m)^{d_{\mathsf{ConEval}}+d+4}$.

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 $\beta_l$. 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 $\gamma$ is 1, to capture the correct decryption, we compute

$$\mu = \left\lceil \mathbf{c}_2 - (\mathbf{c}_0 \| \mathbf{c}_1 \hat{\mathbf{H}}_{s_g \to \bar{r}'} \hat{\mathbf{H}}_{\bar{r},\bar{r}'}) \bar{\mathbf{K}} \right\rfloor_2 = \left\lceil \mu(\mathbf{0} \| \mathbf{g}^T) \cdot \mathsf{BD}(\boldsymbol{\beta}) \otimes \gamma + \bar{\mathbf{e}}_2^T - (\bar{\mathbf{e}}_0^T \| \bar{\mathbf{e}}_1') \bar{\mathbf{K}} \right\rfloor_2,$$

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

Therefore, the norm of error term is bounded by

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

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. □

### 5.3 Security Proof

Now we give the security proof of CAB-PRE scheme, which involves two algorithms $\mathsf{KeyGen}_1$ and $\mathsf{KeyGen}_2$ defined in Section 4.3.

**Theorem 5.** *For a class of functions $\mathcal{F}$ and an inner-product predicate $C_{\boldsymbol{\beta}}$, if $\mathsf{P}$ 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 and its lossy mode.*

*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.
$\mathbf{Game}_1$ : Identical to $\mathbf{Game}_1$ in Section 4.3, except that: the way of generating the matrix $\mathbf{Z}$ is changed by introducing random elements except in the case of $(x \in \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1) \wedge (C_{\boldsymbol{\beta}}(\boldsymbol{\alpha}) = 1)$.
$\mathbf{Game}_2$ : Identical to $\mathbf{Game}_2$ in Section 4.3, except that: the way of generating the matrix $\mathbf{Z}$ is changed by adopting $\mathsf{KeyGen}_1(msk, x)$ algorithm in the case of $(x \in \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1) \wedge (C_{\boldsymbol{\beta}}(\boldsymbol{\alpha}) = 1)$.
$\mathbf{Game}_3$,$\mathbf{Game}_4$ : Identical to $\mathbf{Game}_3$ and $\mathbf{Game}_4$ in Section 4.3, respectively.
$\mathbf{Game}_5$ : Identical to $\mathbf{Game}_4$ in Section 4.3, and the way that generates $\mathbf{u}_3^*$ is also unaltered.
$\mathbf{Game}_6$ : Identical to $\mathbf{Game}_6$ in Section 4.3.
$\mathbf{Game}_7$ : Identical to $\mathbf{Game}_7$ in Section 4.3, except that: the way of generating the matrix $\mathbf{Z}$ is changed by adopting $\mathsf{KeyGen}_2(msk, x)$ algorithm in the case of $(x \in \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1) \wedge (C_{\boldsymbol{\beta}}(\boldsymbol{\alpha}) = 1)$.

$\mathbf{Game}_8$ : Identical to $\mathbf{Game}_8$ in Section 4.3.

$\mathbf{Game}_9$ : Identical to $\mathbf{Game}_9$ in Section 4.3, except that: the challenger chooses $(\mathbf{u}_0^*, \mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{u}_3^*) \in \mathbb{Z}_q^{1\times(m'+ml_f+2l\lceil\log q\rceil)}$ at random. We claim that $|\Pr[S_8]-\Pr[S_9]|$ is negligible for a PPT adversary assuming the hardness of DLWE problem and its lossy mode.

**Reduction from LWE.** Suppose $\mathcal{A}_2$ has a non-negligible advantage in distinguish $\mathbf{Game}_8$ and $\mathbf{Game}_9$. 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 gives $\mathcal{A}_2$ the public parameters $pp = (\mathbf{B}, \mathbf{A}, \mathbf{V}, \mathbf{D}, P.pp)$. The master secret key contains only $\sigma$. 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 queries as in $\mathbf{Game}_9$, except that the challenge query.

- $\mathcal{O}_{\mathsf{Cha}}(f^*, \boldsymbol{\alpha}^*, (\mu_0, \mu_1))$: To generate a challenge ciphertext, $\mathcal{B}$ first picks a random bit $b \in \{0,1\}$, computes $s_{f^*}, \mathbf{u}_1^*, \mathbf{u}_3^*$ as in $\mathbf{Game}_9$ and outputs $ct_{f^*}^2 = (s_{f^*}, \mathbf{u}_0^* = \mathbf{u}_0, \mathbf{u}_1^*, \mathbf{u}_2^* = \mathbf{u}_2' + \mu_b(\mathbf{0}\|\mathbf{g}^T), \mathbf{u}_3^*)$. Then it sets $\mathsf{numCt} := \mathsf{numCt}+1$ and $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. $\mathcal{B}$ adds $ct_{f^*}^2$ in $\mathsf{C}$ with policy tuple $(f^*, \boldsymbol{\alpha}^*, \mathsf{numCt})$ and gives $(\mathsf{numCt}, ct_{f^*}^2)$ to $\mathcal{A}_2$.

**Phase 3 (Decision):** Finally, $\mathcal{A}_2$ guesses if it is interacting with the challenger of $\mathbf{Game}_8$ or $\mathbf{Game}_9$. $\mathcal{B}$ 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 $\mathbf{Game}_8$. Otherwise (*i.e.*, $(\mathbf{u}_0\|\mathbf{u}_2') \xleftarrow{\$} \mathbb{Z}_q^{1\times(m'+l\lceil\log q\rceil)}$), it corresponds to $\mathbf{Game}_9$. Besides, $\mathbf{u}_1^* = \mathbf{u}_0\mathbf{R}\hat{\mathbf{H}}_{msk\to s_{f^*}} + \bar{\mathbf{e}}_1^T$ is uniform and independent in $\mathbb{Z}_q^{1\times ml_f}$ based on a standard application of 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 and its lossy mode, we have $|\Pr[S_8] - \Pr[S_9]| \le \mathsf{negl}(\lambda)$.

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

## 6 Conclusion

In this work, we formalized the notion of CAB-PRE and proposed an adaptively HRA-secure CAB-PRE scheme to enrich the PRE application scenarios. We designed the first adaptively HRA-secure (ciphertext-policy) AB-PRE scheme as a building block. We highlight that this AB-PRE scheme solves the open problem left by Susilo *et al.* [36] in ESORICS '21 about constructing an HRA-secure (ciphertext-policy) AB-PRE scheme. Then, we introduced a well-matched conditional delegation for inner-product predicates based on this AB-PRE scheme to derive our CAB-PRE scheme. Meanwhile, we provided security proof of these two schemes to confirm their security.

We note that key switching 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 problems as an avenue for future work.

## References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: EUROCRYPT. pp. 553–572 (2010)
2. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: CRYPTO. pp. 595–618 (2009)
3. Ateniese, G., Benson, K., Hohenberger, S.: Key-private proxy re-encryption. In: CT-RSA. pp. 279–294 (2009)
4. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: EUROCRYPT. pp. 127–144. Springer (1998)
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: EUROCRYPT. pp. 533–556 (2014)
6. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: ITCS. pp. 309–325 (2012)
7. Brakerski, Z., Vaikuntanathan, V.: Circuit-abe from LWE: unbounded attributes and semi-adaptive security. In: CRYPTO. pp. 363–384 (2016)
8. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: EUROCRYPT. pp. 523–552 (2010)
9. Cohen, A.: What about bob? the inadequacy of CPA security for proxy reencryption. In: PKC. pp. 287–316 (2019)
10. Davidson, A., Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Adaptively secure constrained pseudorandom functions in the standard model. In: CRYPTO. pp. 559–589 (2020)
11. Ge, C., Susilo, W., Baek, J., Liu, Z., Xia, J., Fang, L.: A verifiable and fair attribute-based proxy reencryption scheme for data sharing in clouds. IEEE Transactions on Dependable and Secure Computing **19**(5), 2907–2919 (2022)
12. 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)
13. Ge, C., Susilo, W., Liu, Z., Baek, J., Luo, X., Fang, L.: Attribute-based proxy re-encryption with direct revocation mechanism for data sharing in clouds. IEEE Transactions on Dependable and Secure Computing pp. 1–12 (2023)
14. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC. pp. 197–206 (2008)
15. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: FOCS. pp. 464–479 (1984)
16. Hong, D., Zhaobin, L., Tiezhen, W., Lu, Z.: Certificateless threshold-based conditional proxy re-encryption scheme **46**(1), 44–49,83 (2023)
17. Hu, H., Zhou, Y., Cao, Z., Dong, X.: Efficient and hra secure universal conditional proxy re-encryption for cloud-based data sharing. Applied Sciences **12**(19) (2022)
18. Huang, Q., Yang, Y., Fu, J.: PRECISE: identity-based private data sharing with conditional proxy reencryption in online social networks. Future Gener. Comput. Syst. **86**, 1523–1533 (2018)
19. Katsumata, S., Yamada, S., Yamakawa, T.: Tighter security proofs for GPV-IBE in the quantum random oracle model. In: ASIACRYPT. pp. 253–282 (2018)
20. Li, B., Xu, J., Liu, Y.: Lattice-based fuzzy conditional proxy re-encryption. Journal of Internet Technology **20**(5), 1379–1385 (2019)

21. 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 (2019)
22. Liang, K., Au, M.H., Liu, J.K., Susilo, W., Wong, D.S., Yang, G., Yu, Y., Yang, A.: A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. Future Generation Computer Systems **52**, 95–108 (2015), special Section: Cloud Computing: Security, Privacy and Practice
23. Liang, K., Liu, Z., Tan, X., Wong, D.S., Tang, C.: A cca-secure identity-based conditional proxy re-encryption without random oracles. In: ICISC. pp. 231–246 (2013)
24. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute based proxy re-encryption with delegating capabilities. In: ASIACCS. pp. 276–286 (2009)
25. 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: ESORICS. pp. 147–168 (2021)
26. Liu, Y., Ren, Y., Ge, C., Xia, J., Wang, Q.: A cca-secure multi-conditional proxy broadcast re-encryption scheme for cloud storage system. Journal of Information Security and Applications **47**, 125–131 (2019)
27. Luo, F., Al-Kuwari, S., Wang, F., Chen, K.: Attribute-based proxy re-encryption from standard lattices. Theor. Comput. Sci. **865**, 52–62 (2021)
28. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: EUROCRYPT. pp. 700–718 (2012)
29. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: FOCS. pp. 372–381 (2004)
30. Pareek, G., Purushothama, B.R.: Proxy visible re-encryption scheme with application to e-mail forwarding. In: SIN. pp. 212–217 (2017)
31. Patil, R.Y.: Digital forensics evidence management based on proxy re-encryption. Int. J. Comput. Appl. Technol. **68**(4), 405–413 (2022)
32. Paul, A., Selvi, S.S.D., Rangan, C.P.: Efficient attribute-based proxy re-encryption with constant size ciphertexts. In: INDOCRYPT. pp. 644–665 (2020)
33. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: STOC. pp. 333–342 (2009)
34. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC. pp. 84–93 (2005)
35. 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)
36. Susilo, W., Dutta, P., Duong, D.H., Roy, P.S.: Lattice-based hra-secure attribute-based proxy re-encryption in standard model. In: ESORICS. pp. 169–191 (2021)
37. Tsabary, R.: Fully secure attribute-based encryption for t-cnf from LWE. In: CRYPTO. pp. 62–85 (2019)
38. Wang, C., Han, Y., Duan, X., Guo, K.: Hierarchical identity-based conditional proxy re-encryption scheme based RLWE and NTRU variant. In: ICPCSEE. pp. 240–259 (2021)
39. Weng, J., Deng, R.H., Ding, X., Chu, C., Lai, J.: Conditional proxy re-encryption secure against chosen-ciphertext attack. In: ASIACCS. pp. 322–332 (2009)
40. Weng, J., Yang, Y., Tang, Q., Deng, R.H., Bao, F.: Efficient conditional proxy re-encryption with chosen-ciphertext security. In: Information Security. pp. 151–166 (2009)
41. Yang, Y., Lu, H., Weng, J., Zhang, Y., Sakurai, K.: Fine-grained conditional proxy re-encryption and application. In: ProvSec. pp. 206–222 (2014)

## A Attribute-Based Proxy Re-Encryption

**Syntax**. A unidirectional, single-hop 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 a security parameter $1^\lambda$, 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 message $\mu \in \{0,1\}$, the encryption algorithm outputs a ciphertext $ct_f$ associated with the policy $f$.

$\mathsf{Dec}(sk_x, ct_f) \to \mu/\perp$. 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 an 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 a ciphertext $ct_f$, this algorithm outputs a new ciphertext $ct_g$ associated with the 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 $ct_g \leftarrow \mathsf{ReEnc}(rk_{x\to g}, ct_f)$.

Similar to the HRA security definition of CAB-PRE, Table 4 contains all possible cases of the users' states and their relations in AB-PRE.

Table 4: Correspondences between honest and corrupt users in AB-PRE

| **Relation** | $y \in \mathsf{K} \wedge g(y) = 0$ | $y \in \mathsf{K} \wedge g(y) = 1$ | $y \notin \mathsf{K} \wedge g(y) = 0$ | $y \notin \mathsf{K} \wedge g(y) = 1$ |
|---|---|---|---|---|
| $x \in \mathsf{K} \wedge f(x) = 0$ | H → H | H → C | H → H | H → H |
| $x \in \mathsf{K} \wedge f(x) = 1$ | C → H | C → C | C → H | C → H |
| $x \notin \mathsf{K} \wedge f(x) = 0$ | H → H | H → C | H → H | H → H |
| $x \notin \mathsf{K} \wedge f(x) = 1$ | H → H | H → C | H → H | H → H |

**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 $\mathsf{K}$. It gives $sk_x$ to $\mathcal{A}$.
- $\mathcal{O}_{\mathsf{Enc}}(f, \mu)$: For an encryption query $(f, \mu)$, the challenger computes $ct_f \leftarrow \mathsf{Enc}(f, \mu)$, sets $\mathsf{numCt} := \mathsf{numCt} + 1$, adds $ct_f$ in $\mathsf{C}$ with policy tuple $(f, \mathsf{numCt})$, and gives $(\mathsf{numCt}, ct_f)$ to $\mathcal{A}$.
- $\mathcal{O}_{\mathsf{ReKey}}(x, g)$: For a re-encryption key query $(x, g)$, if exist $y \in \mathsf{K}$ and $(f, \cdot) \in \mathsf{C}$ such that 1) $(x \in \mathsf{K} \wedge f(x) = 0) \wedge (y \in \mathsf{K} \wedge g(y) = 1)$, or 2) $(x \notin \mathsf{K} \wedge f(x) = 0) \wedge (y \in \mathsf{K} \wedge g(y) = 1)$, or $(x \notin \mathsf{K} \wedge f(x) = 1) \wedge (y \in \mathsf{K} \wedge g(y) = 1)$ holds, the challenger returns $\perp$; otherwise, it generates $rk_{x\to g} \leftarrow \mathsf{ReKeyGen}(sk_x, g)$ and gives it 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 $\mathsf{C}$ with policy tuple $(f^*, \mathsf{numCt})$ 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 $k \leq \mathsf{numCt}$. The challenger does the following operations.
  1) If there is no value in $\mathsf{C}$ with policy tuple $(f, k)$, return $\perp$.
  2) If $f(x) = 0$, return $\perp$.
  3) If exists $y \in \mathsf{K}$ such that $(y \in \mathsf{K} \wedge g(y) = 1) \wedge k \in \mathsf{Deriv}$ holds, return $\perp$.
  4) Otherwise, let $ct_f$ be that value in the store $\mathsf{C}$. 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 $\mathsf{C}$ with policy tuple $(g, \mathsf{numCt})$. If $k \in \mathsf{Deriv}$, set $\mathsf{Deriv} := \mathsf{Deriv} \cup \{\mathsf{numCt}\}$. Finally, it gives $(\mathsf{numCt}, ct_g)$ 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 adaptively HRA-secure if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ winning in the game is negligible.