Adaptively Secure Inner Product Encryption from LWE

Shuichi Katsumata[1], Ryo Nishimaki[2], Shota Yamada[1], and Takashi Yamakawa[2(✉)]

[1] AIST, Tokyo, Japan
{shuichi.katsumata,yamada-shota}@aist.go.jp
[2] NTT Secure Platform Laboratories, Tokyo, Japan
{ryo.nishimaki.zk,takashi.yamakawa.ga}@hco.ntt.co.jp

Abstract. Attribute-based encryption (ABE) is an advanced form of encryption scheme allowing for access policies to be embedded within the secret keys and ciphertexts. By now, we have ABEs supporting numerous types of policies based on hardness assumptions over bilinear maps and lattices. However, one of the distinguishing differences between ABEs based on these two breeds of assumptions is that the former can achieve *adaptive* security for quite expressible policies (e.g., inner-products, boolean formula) while the latter can not. Recently, two adaptively secure lattice-based ABEs have appeared and changed the state of affairs: a non-zero inner-product (NIPE) encryption by Katsumata and Yamada (PKC'19) and an ABE for t-CNF policies by Tsabary (CRYPTO'19). However, the policies supported by these ABEs are still quite limited and do not embrace the more interesting policies that have been studied in the literature. Notably, constructing an adaptively secure *inner-product encryption* (IPE) based on lattices still remains open.

In this work, we propose the first adaptively secure IPE based on the learning with errors (LWE) assumption with sub-exponential modulus size (without resorting to complexity leveraging). Concretely, our IPE supports inner-products over the integers $\mathbb{Z}$ with polynomial sized entries and satisfies adaptively weakly-attribute-hiding security. We also show how to convert such an IPE to an IPE supporting inner-products over $\mathbb{Z}_p$ for a polynomial-sized p and a fuzzy identity-based encryption (FIBE) for small and large universes. Our result builds on the ideas presented in Tsabary (CRYPTO'19), which uses constrained pseudorandom functions (CPRF) in a semi-generic way to achieve adaptively secure ABEs, and the recent lattice-based adaptively secure CPRF for inner-products by Davidson et al. (CRYPTO'20). Our main observation is realizing how to weaken the *conforming* CPRF property introduced in Tsabary (CRYPTO'19) by taking advantage of the specific linearity property enjoyed by the lattice evaluation algorithms by Boneh et al. (EUROCRYPT'14).

1 Introduction

An attribute-based encryption (ABE) [44] is an advanced form of public-key encryption (PKE) that allows the sender to specify in a more general way about

S. Moriai and H. Wang (Eds.): ASIACRYPT 2020, LNCS 12493, pp. 375–404, 2020.
https://doi.org/10.1007/978-3-030-64840-4_13

who should be able to decrypt. In an ABE for predicate $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, decryption of a ciphertext associated with an attribute $\mathbf{y}$ is only possible by a secret key associated with an attribute $\mathbf{x}$ such that $\mathsf{P}(\mathbf{x},\mathbf{y}) = 1$. For instance, identity-based encryption (IBE) [9,22] is a special form of ABE where an equality predicate is considered.

Over the past decade and a half, we have seen exciting progress in the design and security analysis of ABEs. Each subsequent work provides improvements in various aspects including security, expressiveness of predicates, or underlying assumptions. While the earlier constructions were mainly based on bilinear maps, e.g., [8,11,32,34,44,45], by now we have plenty of constructions based on lattices as well, e.g., [1,3,10,19,25,28]. Some of the types of ABEs that have attracted more attention than others in the literature include (but not limited to), fuzzy IBE [2,44], inner-product encryption (IPE) [3,34,36], ABE for boolean formulae [32,36], and ABE for P/poly circuits [10,28]. Regarding the expressiveness of predicates, lattice-based ABEs seem to achieve stronger results than bilinear map-based ABEs since the former allows for predicates expressible by P/poly circuits, whereas the latter is restricted to boolean formulae.

Adaptive Security. While lattice-based ABEs have richer expressiveness, bilinear map-based ABEs can realize stronger security. Specifically, they can address *adaptive* security (in the standard model) for quite expressive predicates. Here, adaptive security states that, even if an adversary can obtain polynomially many secret keys for any attribute $\mathbf{x}$ and adaptively query for a challenge ciphertext associated with an attribute $\mathbf{y}^*$ such that $\mathsf{P}(\mathbf{x},\mathbf{y}^*) = 0$, it still cannot learn the message encrypted within the challenge ciphertext. This clearly captures the real-life scenario where an adversary can adaptively choose which attributes to attack. In some cases, we may consider the much weaker *selective* security, where an adversary must declare which attribute $\mathbf{y}^*$ it will query as the challenge at the beginning of the security game. In general, we can convert a selectively secure scheme to an adaptively secure scheme by employing complexity leveraging, where the reduction algorithm simply guesses the challenge attribute at the outset of the game. However, this is often undesirable as such proofs incur an exponential security loss and necessitate in relying on exponentially hard assumptions. Using bilinear maps, we know how to directly construct adaptively secure fuzzy IBE [20,49], IPE [20,36,38,49], and even ABE for boolean formulae [5,6,20,36,39,49] from standard (polynomial) assumptions.

On the other hand, our knowledge of adaptively secure lattice-based ABEs is still quite limited. Notably, most of the lattice-based ABEs are only selectively secure. For almost a decade, the only adaptively secure scheme we knew how to construct from lattices was limited to the most simplistic form of ABE, an IBE [1,19]. Considering that we had a lattice-based selectively secure ABE for the powerful predicate class of P/poly circuits, this situation on adaptive security was unsatisfactory. Recently, the state of affairs changed: Katsumata and Yamada [33] proposed an adaptively secure non-zero IPE (NIPE), and Tsabary [46] proposed an adaptively secure ABE for t-CNF predicates. The latter predicate consists of formulas in conjunctive normal form where each clause depends

on at most t bits of the input, for any constant t. The former work is based on a generic construction from adaptively secure functional encryption for inner-products [4], whereas the latter work ingeniously extends the adaptively secure bilinear map-based IBE of Gentry [24] to the lattice setting by utilizing a special type of *constrained* pseudorandom function (CPRF) [12,13,35]. Unfortunately, NIPE nor ABE for t-CNF is not expressive enough to capture the more interesting types of ABE such as fuzzy IBE or IPE, let al.one ABE for boolean formulae or $\mathsf{P/poly}$. Therefore, the gap between the bilinear map setting and the lattice setting regarding adaptive security still remains quite large and dissatisfying. Indeed, constructing an adaptively secure IPE based on lattices is widely regarded as one of the long-standing open problems in lattice-based ABE.

1.1 Our Contribution

In this work, we propose the first lattice-based *adaptively* secure IPE over the integers $\mathbb{Z}$. In addition, we show several extensions of our main result to realize other types of ABEs such as fuzzy IBE. The results are summarized below and in Table 1. All of the following schemes are secure under the learning with errors (LWE) assumption with sub-exponential modulus size.

- We construct an adaptively secure IPE over the integers ($\mathbb{Z}$) with polynomial sized entries. The predicate is defined as $\mathsf{P} : \mathcal{Z} \times \mathcal{Z} \to \{0,1\}$, where $\mathcal{Z}$ is a subset of $\mathbb{Z}^\ell$ with bounded polynomial sized entries and $\mathsf{P}(\mathbf{x},\mathbf{y}) = 1$ if and only if $\langle \mathbf{x},\mathbf{y}\rangle = 0$ over $\mathbb{Z}$.
- We construct an adaptively secure IPE over the ring $\mathbb{Z}_p$ for $p = \mathsf{poly}(\kappa)$. The predicate $\mathsf{P}_{\mathsf{mod}} : \mathbb{Z}_p^\ell \times \mathbb{Z}_p^\ell \to \{0,1\}$ is defined similarly to above, where now $\mathsf{P}_{\mathsf{mod}}(\mathbf{x},\mathbf{y}) = 1$ if and only if $\langle \mathbf{x},\mathbf{y}\rangle = 0 \mod p$.
- We construct an adaptively secure fuzzy IBE for small and large universe with threshold T. Specifically, the predicate is defined as $\mathsf{P}_{\mathsf{fuz}} : \mathcal{D}^n \times \mathcal{D}^n \to \{0,1\}$, where $\mathcal{D}$ is a set of either polynomial size (i.e., small universe) or exponential size (i.e., large universe) and $\mathsf{P}_{\mathsf{fuz}}(\mathbf{x},\mathbf{y}) = 1$ if and only if $\mathsf{HD}(\mathbf{x},\mathbf{y}) \leq n - T$. Here, HD denotes the hamming distance. That is, if $\mathbf{x}$ and $\mathbf{y}$ are identical in more than T-positions, then $\mathsf{P}_{\mathsf{fuz}}(\mathbf{x},\mathbf{y}) = 1$.

Though we mainly focus on proving payload-hiding for these constructions, we can generically upgrade payload-hiding ABE to be weakly-attribute-hiding by using lockable obfuscation, which is known to exist under the LWE assumption with sub-exponential modulus size [31,50]. Therefore, we obtain adaptively weakly-attribute-hiding ABE for the above classes of predicates under the LWE assumption with sub-exponential modulus size. We note that this does not require an additional assumption since our payload-hiding constructions already rely on the same assumption.

The first construction is obtained by extending the recent result by Tsabary [46], while the second and third constructions are obtained by a generic transformation of the first construction.

Table 1. Existing adaptively secure lattice-based ABE.

Reference	Type of predicate	LWE Asmp.
ABB'10 [1], CHKP'10 [19]	IBE and HIBE w/ $O(1)$-hierarchy	poly
KY'19 [33]	$\mathbb{Z}$ w/ poly-size entries	poly
	NIPE over $\mathbb{Z}$ w/ exp-size entries	poly
	$\mathbb{Z}_p$ w/ poly and exp-size p †	subexp
Tsabary'19 [46]	(CP-)ABE for t-CNF where $t = O(1)$	subexp
Ours	IPE over $\mathbb{Z}$ w/ poly-size entries	subexp
Ours	IPE over $\mathbb{Z}_p$ w/ poly-size p	subexp
Ours	Fuzzy IBE w/ small and large universe	subexp

† : The key generation algorithm is stateful for NIPE over $\mathbb{Z}_p$.

1.2 Technical Overview

We provide a detailed overview of our first (main) result regarding an adaptively secure IPE over the integers ($\mathbb{Z}$) and provide some discussions on how to extend it to ABE with other types of useful predicates. For our first result, we first extend the framework of Tsabary [46] and exploit a specific linearity property of the lattice evaluation algorithms of Boneh et al. [10]. We then make a subtle (yet crucial) modification to the CPRF for inner-products over the integer by Davidson et al. [23] so as to be compatible with our extended framework for achieving adaptively secure ABEs.

Note. In the following, to make the presentation clearer, we treat ABE as either a *ciphertext-policy* (CP) ABE or a *key-policy* (KP) ABE interchangeably. In CP-ABE, an attribute associated to a ciphertext represents a policy $f \in \mathcal{Y}$, which is described as a circuit, and we define the predicate $\mathsf{P}(\mathbf{x}, f) := f(\mathbf{x})$. That is, the predicate is satisfied if $f(\mathbf{x}) = 1$. KP-ABE is defined analogously. Note that IPE can be viewed as both a CP and KP-ABE since the roles of the attributes associated with the secret key and the ciphertext are symmetric.

Reviewing Previous Results. Due to the somewhat lattice-heavy nature of our result, we review the relevant known results. For those who are up-to-date with the result of Tsabary [46] may safely skip to "Our Results". We first provide some background on lattice evaluation algorithms [10]. We then review the framework developed by Tsabary [46] for achieving adaptively secure ABEs (for t-CNF).

***Selectively* Secure (KP-)ABE Based on Homomorphic Evaluation.** We recall the selectively secure ABE by Boneh et al. [10], which is the basic recipe for constructing lattice-based ABEs. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m}$ be a public matrix and $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ be the so-called (public) gadget matrix whose trapdoor is known [37]. Then, there exists two deterministic efficiently computable lattice evaluation algorithms $\mathsf{PubEval}$ and CtEval such that for any $f : \{0,1\}^\ell \to \{0,1\}$ and $\mathbf{x} \in \{0,1\}^\ell$, the following property holds.[1]

– $\mathsf{PubEval}(f, \mathbf{A}) \to \mathbf{A}_f$,

[1] We note that f can also be represented as an arithmetic circuit.

– $\mathsf{CtEval}(f, \mathbf{x}, \mathbf{A}, \mathbf{s}^\top(\mathbf{A} - \mathbf{x}^\top \otimes \mathbf{G}) + \mathsf{noise}) \to \mathbf{s}^\top(\mathbf{A}_f - f(\mathbf{x}) \otimes \mathbf{G}) + \mathsf{noise}$,

where noise denotes some term whose size is much smaller than q which we can ignore. In words, CtEval is an algorithm that allows to convert a ciphertext (or an encoding) of $\mathbf{x}$ w.r.t. matrix $\mathbf{A}$ into a ciphertext of $f(\mathbf{x})$ w.r.t. matrix $\mathbf{A}_f$, where $\mathbf{A}_f$ is the same matrix output by PubEval. In the following, we assume that the output of CtEval statistically hides the value $\mathbf{x}$, which is possible by adding sufficiently large noise.

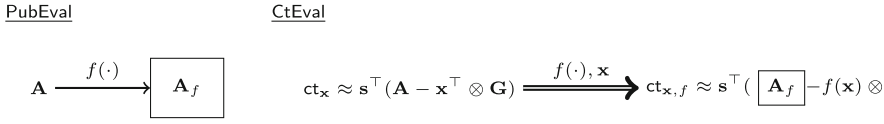

Fig. 1. PubEval and CtEval. In all figures, symbol $\approx$ means that we hide (or ignore) the noise part in ciphertexts.

We provide an overview of how to construct a (KP-)ABE. The public parameters consist of a matrix $\mathbf{A}$ and a vector $\mathbf{u}$. Let $\hat{f}$ be a negation of the function f, that is, $\hat{f}(\mathbf{x}) := 1 - f(\mathbf{x})$. To generate a secret key for function f, the KeyGen algorithm first runs $\mathbf{A}_{\hat{f}} \leftarrow \mathsf{PubEval}(\hat{f}, \mathbf{A})$ as in Equation (1) below. Then the secret key sk_f is sampled as a short vector $\mathbf{e}_f$ such that $\mathbf{A}_{\hat{f}}\mathbf{e}_f = \mathbf{u}$.[2] To generate a ciphertext for attribute $\mathbf{x}$ with message $\mathsf{M} \in \{0, 1\}$, the Enc algorithm generates a LWE sample of the form $\mathsf{ct}_0 := \mathbf{s}^\top\mathbf{u} + \mathsf{noise} + \mathsf{M} \cdot \lfloor q/2 \rfloor$ and $\mathsf{ct}_\mathbf{x}$ as depicted on the l.h.s. of Equation (2). To decrypt with a secret key sk_f, the Dec algorithm first runs $\mathsf{CtEval}(\hat{f}, \mathbf{x}, \mathbf{A}, \mathsf{ct}_\mathbf{x})$ to generate $\mathsf{ct}_{\mathbf{x},\hat{f}}$ as depicted on the r.h.s. of Equation (2). Here, notice that the ciphertext is converted into a ciphertext that encodes the matrix $\mathbf{A}_{\hat{f}}$ used during KeyGen (both boxed in Equations (1) and (2)). Then, if the predicate is satisfied, i.e., $f(\mathbf{x}) = 1 \Leftrightarrow \hat{f}(\mathbf{x}) = 0$, then $\mathsf{ct}_{\mathbf{x},f} = \mathbf{s}^\top\mathbf{A}_{\hat{f}} + \mathsf{noise}$. Therefore, using $\mathbf{e}_f$, the message can be recovered by computing $\mathsf{ct}_0 - \langle \mathsf{ct}_{\mathbf{x},f}, \mathbf{e}_f \rangle$ and rounding appropriately.

Now, *selective* security follows by embedding the LWE problem in the challenge ciphertext. Specifically, the reduction algorithm is given an LWE instance $([\mathbf{u}|\mathbf{B}], [\mathbf{v}_0|\mathbf{v}])$, where $[\mathbf{v}_0|\mathbf{v}]$ is either random or of the form $[\mathbf{v}_0|\mathbf{v}] = \mathbf{s}^\top[\mathbf{u}|\mathbf{B}] + \mathsf{noise}$. It then implicitly sets $\mathbf{A} := \mathbf{B}\mathbf{R} + \mathbf{x}^{*\top} \otimes \mathbf{G}$ where $\mathbf{x}^*$ is the challenge attribute the adversary commits to at the outset of the security game and $\mathbf{R}$ is a random matrix with small entries and sets the challenge ciphertext as $(\mathsf{ct}_0 := \mathbf{v}_0 + \mathsf{M} \cdot \lfloor q/2 \rfloor, \mathsf{ct}_{\mathbf{x}^*} := \mathbf{v})$. It can be checked that if $[\mathbf{v}_0|\mathbf{v}]$ is a valid LWE instance, then the challenge is distributed as in the actual security game. Otherwise, the challenge ciphertext is uniformly random. Finally, we remark that simulating secret keys for policy f such that $f(\mathbf{x}^*) = 0$ is possible since there exists

[2] To be accurate, we require an extra matrix $\mathbf{A}_0$ for which we know a trapdoor in order to sample such a short vector. However, we simplify the exposition for the sake of clarity.

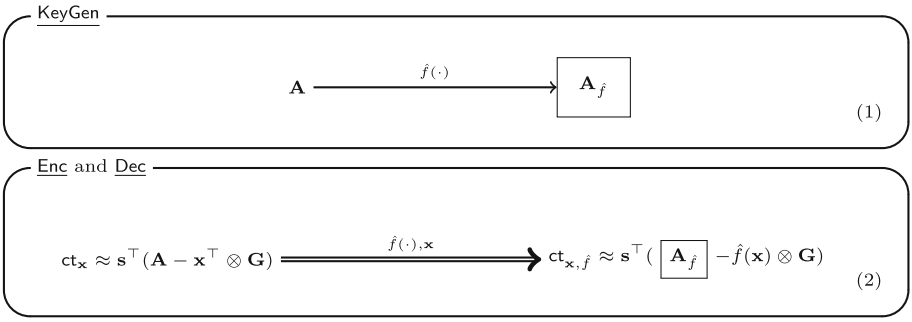

Fig. 2. Illustration of the selectively secure ABE by BGG+14. The thin (resp. thick) black arrow describes running algorithm PubEval (resp. CtEval). The items on top of the arrows denote the required input to run the respective algorithms. This is the same for all subsequent figures. In Equation (2), the l.h.s. and r.h.s. are generated by Enc and Dec, respectively.

a special lattice evaluation algorithm (only used during the security proof) that allows the reduction algorithm to convert $\mathbf{A}_{\hat{f}}$ into $\mathbf{B}\mathbf{R}_{\hat{f}} + \hat{f}(\mathbf{x}^*) \otimes \mathbf{G} = \mathbf{B}\mathbf{R}_{\hat{f}} + \mathbf{G}$, where $\mathbf{R}_{\hat{f}}$ is a matrix with short norm. We omit the details on what or how to use $\mathbf{R}_{\hat{f}}$ as it is not important for this overview and refer the readers to [10].

We end by emphasizing that the above reduction technique only works in the selective setting because the adversary commits to $\mathbf{x}^*$ at the outset of the game; if it did not, then the reduction algorithm will not be able to set $\mathbf{A}$ as $\mathbf{B} + \mathbf{x}^{*\top} \otimes \mathbf{G}$ in the public parameter.

Adaptively Secure IBE à la Gentry [24] and Tsabary [46].[3] Before getting into adaptively secure ABEs, we first consider the simpler adaptively secure IBEs. We overview the so-called "tagging" technique [24,46]. In the real scheme, a secret key and a ciphertext for an identity id are associated with random "tags" r_{id}. The scheme is set up so that decryption only works if the tag value r_{id} of the secret key $\mathsf{sk}_{\mathsf{id}}$ is different from the tag value $\tilde{r}_{\mathsf{id}}$ of the ciphertext for an identity id. In case the tags are sampled from an exponentially large space, such a scheme only has a negligible probability of a decryption failure. At a high level, the scheme will be tweaked so that the reduction algorithm assigns exactly one random tag r_{id} per identity id; a secret key and a challenge ciphertext for the same identity id are tagged by the same r_{id}. In addition, the reduction algorithm will only be able to simulate a secret key and a challenge ciphertext w.r.t. this unique tag r_{id}. Here, this tweak will remain unnoticed by the adversary since a valid adversary never asks for a secret key and a challenge ciphertext for the same identity id.

We briefly review how Tsabary [46] cleverly carried out this idea in the lattice-setting. The public parameter now includes a description of a pseudo-

[3] One can also see this construction as an analogy of Waters' dual system framework [48].

random function PRF, and the master secret key includes a seed k for the PRF. To generate a secret key for identity id, the KeyGen algorithm computes the random tag $r_{\mathsf{id}} \leftarrow \mathsf{PRF.Eval}(\mathsf{k}, \mathsf{id})$. It then sequentially runs $\mathbf{A}^{\mathsf{eval}}_{\mathsf{id}} \leftarrow \mathsf{PubEval}(\mathsf{PRF.Eval}(\cdot, \mathsf{id}), \mathbf{A})$ and $\mathbf{A}^{\mathsf{eq}}_{\mathsf{id},r_{\mathsf{id}}} \leftarrow \mathsf{PubEval}(\mathsf{Eq}_{r_{\mathsf{id}}}(\cdot), \mathbf{A}^{\mathsf{eval}}_{\mathsf{id}})$ as in Equation (3) below, where $\mathsf{Eq}_{r_{\mathsf{id}}}(\tilde{r}_{\mathsf{id}}) = 1$ if and only if $r_{\mathsf{id}} = \tilde{r}_{\mathsf{id}}$. As before, it then samples a short vector $\mathbf{e}_{\mathsf{id}}$ such that $\mathbf{A}^{\mathsf{eq}}_{\mathsf{id},r_{\mathsf{id}}}\mathbf{e}_{\mathsf{id}} = \mathbf{u}$. The final secret key is $\mathsf{sk}_{\mathsf{id}} := (r_{\mathsf{id}}, \mathbf{e}_{\mathsf{id}})$. To generate a ciphertext for identity id with message M, the Enc algorithm first samples a random PRF key $\widetilde{\mathsf{k}}$ and generates $\mathsf{ct}_0 := \mathbf{s}^\top\mathbf{u} + \mathsf{noise} + \mathsf{M}\cdot\lfloor q/2 \rfloor$ as before. It then generates $\mathsf{ct}_{\widetilde{\mathsf{k}}}$ as depicted in the l.h.s of Equation (4) and further executes $\mathsf{ct}^{\mathsf{eval}}_{\mathsf{id}} \leftarrow \mathsf{CtEval}(\mathsf{PRF.Eval}(\cdot, \mathsf{id}), \widetilde{\mathsf{k}}, \mathbf{A}, \mathsf{ct}_{\widetilde{\mathsf{k}}})$ as depicted in the r.h.s of Equation (4). The final ciphertext is $\mathsf{ct} := (\widetilde{r}_{\mathsf{id}}, \mathsf{ct}_0, \mathsf{ct}^{\mathsf{eval}}_{\mathsf{id}})$, where $\widetilde{r}_{\mathsf{id}} \leftarrow \mathsf{PRF.Eval}(\widetilde{\mathsf{k}}, \mathsf{id})$. Effectively, the Enc algorithm has constructed a ciphertext that is bound to an identity id and a random tag $\tilde{r}_{\mathsf{id}}$; observe that $\mathbf{A}^{\mathsf{eval}}_{\mathsf{id}}$ is the same matrix that appears during KeyGen (in a single-framed box). Here, we note that the noise term in $\mathsf{ct}_{\mathsf{id}}$ does not leak any information on the PRF key $\widetilde{\mathsf{k}}$ by our assumption. Now, to decrypt, the Dec algorithm, with knowledge of both the random tag r_{id} and $\tilde{r}_{\mathsf{id}}$, runs $\mathsf{ct}^{\mathsf{eq}}_{\mathsf{id},r_{\mathsf{id}}} \leftarrow \mathsf{CtEval}(\mathsf{Eq}_{r_{\mathsf{id}}}(\cdot), \tilde{r}_{\mathsf{id}}, \mathbf{A}^{\mathsf{eval}}_{\mathsf{id}}, \mathsf{ct}^{\mathsf{eval}}_{\mathsf{id}})$ as depicted in the r.h.s. of Equation (5). At this point, the ciphertext is converted into a ciphertext that encodes the matrix $\mathbf{A}^{\mathsf{eq}}_{\mathsf{id},r_{\mathsf{id}}}$ used during KeyGen (in a double-framed box), and we have $\mathsf{Eq}_{r_{\mathsf{id}}}(\tilde{r}_{\mathsf{id}}) = 0$ since $r_{\mathsf{id}} \neq \tilde{r}_{\mathsf{id}}$ with all but a negligible probability. Hence, since $\mathsf{ct}^{\mathsf{eq}}_{\mathsf{id},r_{\mathsf{id}}} = \mathbf{s}^\top\mathbf{A}^{\mathsf{eq}}_{\mathsf{id},r_{\mathsf{id}}} + \mathsf{noise}$, the Dec algorithm can decrypt the ciphertext using the short vector $\mathbf{e}_{\mathsf{id}}$ included in the secret key following the same argument as before.

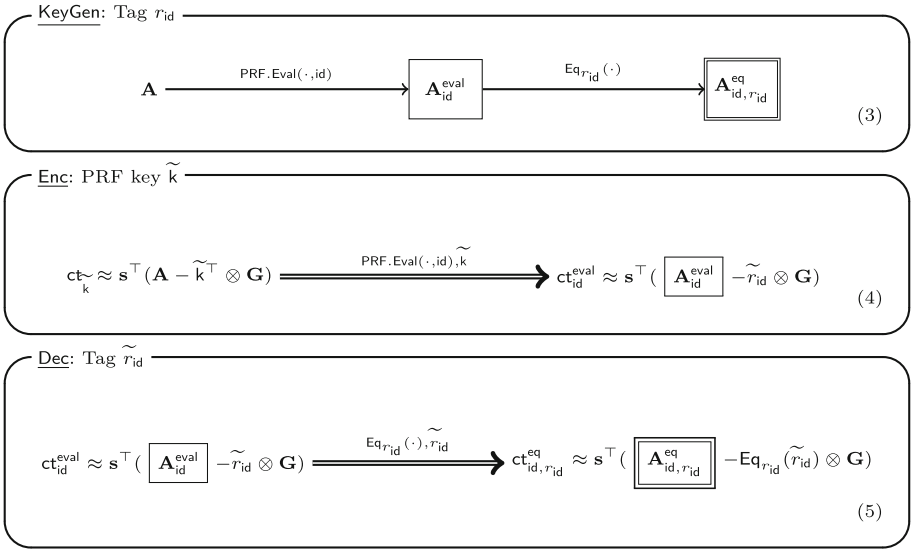

Fig. 3. Illustration of the adaptively secure IBE by Tsabary.

The key observation is that a ciphertext for an identity id is generated from $\mathsf{ct}_{\widetilde{\mathsf{k}}}$ that *only depends on the* PRF *key*. Notably, *adaptive* security can be achieved (informally) because the reduction algorithm no longer needs to guess the challenge identity id and by the *adaptive* pseudorandomness of the PRF. We provide a proof sketc.h to get a better intuition for the more complex subsequent ABE construction: We first modify the security game so that the challenger no longer needs to explicitly embed $\widetilde{\mathsf{k}}$ in the ciphertext. Namely, the challenger simply computes $\mathbf{A}_{\mathsf{id}}$ using $\mathsf{PubEval}$, which it can run without knowledge of $\widetilde{\mathsf{k}}$, and directly generates $\mathsf{ct}_{\mathsf{id}}^{\mathsf{eval}}$ using $\tilde{r}_{\mathsf{id}}$. This is statistically the same as in the real scheme since the noise term statistically hides $\widetilde{\mathsf{k}}$ due to the assumption. Now, we can invoke the adaptive pseudorandomness of the PRF. The reduction algorithm generates the random tag associated with the challenge ciphertext by implicitly using the seed k included in the master secret key (by querying its own PRF challenger) instead of sampling a fresh $\widetilde{\mathsf{k}}$. Note that the random tag associated with the secret key and challenge ciphertext for the same id are identical now. We then switch back to the real scheme where the Enc algorithm first constructs ct_{k}, where the only difference is that k is encoded rather than a random PRF seed $\widetilde{\mathsf{k}}$. At this point, we can rely on the same argument as the selective security of [10] since k is known at the outset of the game and the reduction algorithm (which is the LWE adversary) can set $\mathbf{A} := \mathbf{B} + \mathsf{k}^{\top} \otimes \mathbf{G}$. The challenge ciphertext for any id^* can be computed by simply running CtEval on $\mathsf{ct}_{\mathsf{k}} = \mathbf{v}$, where $\mathbf{v} = \mathbf{s}^{\top}\mathbf{B} + \mathsf{noise}$ for a valid LWE instance. In addition, a secret key for any id can be simulated as well since we have $\mathbf{A}_{\mathsf{id},r_{\mathsf{id}}}^{\mathsf{eq}} = \mathbf{B}\mathbf{R}_{\mathsf{id},r_{\mathsf{id}}}^{\mathsf{eq}} + \mathsf{Eq}_{r_{\mathsf{id}}}(r_{\mathsf{id}}) \otimes \mathbf{G} = \mathbf{B}\mathbf{R}_{\mathsf{id},r_{\mathsf{id}}}^{\mathsf{eq}} + \mathbf{G}$ for a matrix $\mathbf{R}_{\mathsf{id},r_{\mathsf{id}}}^{\mathsf{eq}}$ with low norm.

Adaptively Secure (CP-)ABE Using (conforming) Constrained PRF. Tsabary [46] made the keen observation of using a *CPRF* instead of a standard PRF in the above idea to construct an ABE. A CPRF allows a user to learn *constrained keys* to evaluate the PRF only on inputs $\mathbf{x}$ satisfied by a constraint f. Let k be the secret key (i.e., seed) to the "base" PRF. Algorithm $\mathsf{CPRF.Eval}$ takes as an input k and $\mathbf{x}$ and outputs a random value $r_{\mathbf{x}}$ as a standard PRF. Algorithm $\mathsf{CPRF.Constrain}$ takes as input k and a constraint f, represented as a circuit, and outputs a constrained key $\mathsf{k}_f^{\mathsf{con}}$. Then, algorithm $\mathsf{CPRF.ConstrainEval}$ takes as input $\mathsf{k}_f^{\mathsf{con}}$ and $\mathbf{x}$ and outputs $r'_{\mathbf{x}}$, where $r'_{\mathbf{x}} = r_{\mathbf{x}}$ if the input is satisfied by the constraint, i.e., $f(\mathbf{x}) = 1$. Now (adaptive) pseudorandomness of a CPRF stipulates that even if an adversary can adaptively query $\mathsf{CPRF.Eval}(\mathsf{k}, \cdot)$ on any input of its choice and receive a constrained key $\mathsf{k}_f^{\mathsf{con}}$ for any constraint f, the value $\mathsf{CPRF.Eval}(\mathsf{k}, \mathbf{x}^*)$ remains pseudorandom to the adversary as long as $f(\mathbf{x}^*) = 0$.

We now explain an initially flawed but informative approach of plugging in a CPRF in the above idea to construct a (CP-)ABE and explain how Tsabary [46] overcomes it. The master secret key for the ABE now includes the secret key k for the CPRF. To generate a secret key for an attribute $\mathbf{x}$, the KeyGen algorithm first computes a random tag $r_{\mathbf{x}} \leftarrow \mathsf{CPRF.Eval}(\mathsf{k}, \mathbf{x})$. It then sequentially runs $\mathbf{A}_{\mathbf{x}}^{\mathsf{eval}} \leftarrow \mathsf{PubEval}(\mathsf{CPRF.Eval}(\cdot, \mathbf{x}), \mathbf{A})$ and $\mathbf{A}_{\mathbf{x},r_{\mathbf{x}}}^{\mathsf{eq}} \leftarrow \mathsf{PubEval}(\mathsf{Eq}_{r_{\mathbf{x}}}(\cdot), \mathbf{A}_{\mathbf{x}}^{\mathsf{eval}})$ as in Equation (6) below. Finally, a short vector $\mathbf{e}_{\mathbf{x}}$ such that $\mathbf{A}_{\mathbf{x},r_{\mathbf{x}}}^{\mathsf{eq}}\mathbf{e}_{\mathbf{x}} = \mathbf{u}$ is sampled. The

final secret key is $\mathsf{sk}_{\mathbf{x}} := (r_{\mathbf{x}}, \mathbf{e}_{\mathbf{x}})$. To encrypt with respect to a policy f, the Enc algorithm prepares a constrained key for f, which will later be used to derive random tags for any $\mathbf{x}$ during decryption. Specifically, it first samples a fresh secret key $\widetilde{\mathsf{k}}$ for the CPRF and generates $\mathsf{ct}_0 := \mathbf{s}^\top \mathbf{u} + \mathsf{noise} + \mathsf{M} \cdot \lfloor q/2 \rfloor$ as before. It then generates $\mathsf{ct}_{\widetilde{\mathsf{k}}}$ and further executes $\mathsf{ct}_f^{\mathsf{con}} \leftarrow \mathsf{CtEval}(\mathsf{CPRF.Constrain}(\cdot, f), \widetilde{\mathsf{k}}, \mathbf{A}, \mathsf{ct}_{\widetilde{\mathsf{k}}})$ as depicted in Equation (7). The final ciphertext is $\mathsf{ct} := (\widetilde{\mathsf{k}}_f^{\mathsf{con}}, \mathsf{ct}_0, \mathsf{ct}_f^{\mathsf{con}})$, where $\widetilde{\mathsf{k}}_f^{\mathsf{con}} \leftarrow \mathsf{CPRF.Constrain}(\widetilde{\mathsf{k}}, f)$ is a constrained key and note that $\mathsf{ct}_f^{\mathsf{con}}$ statistically hides the information on $\widetilde{\mathsf{k}}$. Observe that the ciphertext encodes the policy f.

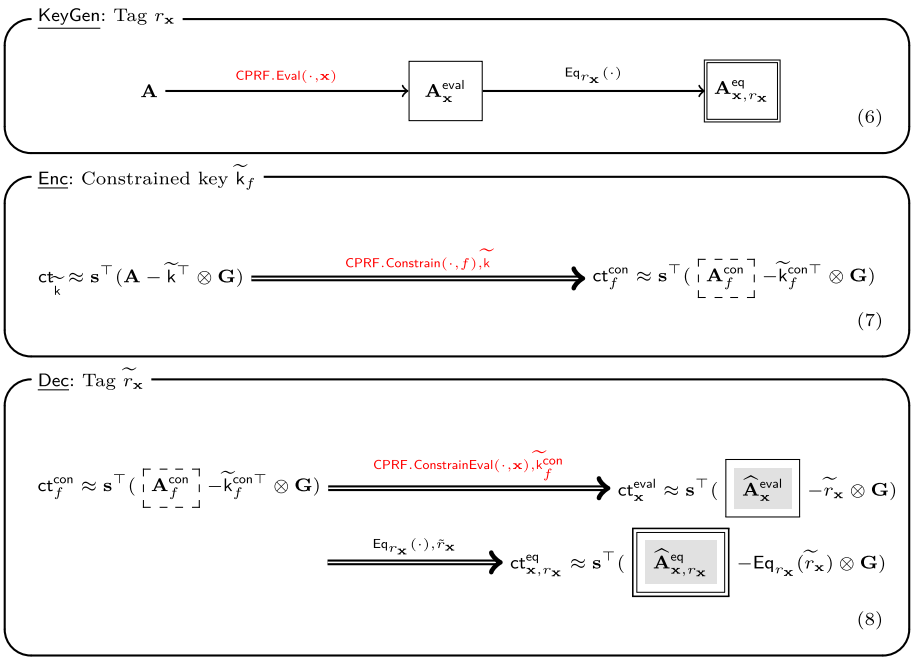

Fig. 4. Illustration of the high-level structure of the adaptively secure CP-ABE by Tsabary.

However, at this point, the problem becomes apparent: Decryption no longer works. What the decryptor in possession of secret key $\mathsf{sk}_{\mathbf{x}}$ can do is to convert the ciphertext $\mathsf{ct}_f^{\mathsf{con}}$ into $\mathsf{ct}_{\mathbf{x}}^{\mathsf{eval}} \leftarrow \mathsf{CtEval}(\mathsf{CPRF.ConstrainEval}(\cdot, \mathbf{x}), \widetilde{\mathsf{k}}_f^{\mathsf{con}}, \mathbf{A}_f^{\mathsf{con}}, \mathsf{ct}_f^{\mathsf{con}})$ as depicted in Equation (8). In addition, it can further convert it into $\mathsf{ct}_{\mathbf{x}, r_{\mathbf{x}}}^{\mathsf{eq}} \leftarrow \mathsf{CtEval}(\mathsf{Eq}_{r_{\mathbf{x}}}(\cdot), \tilde{r}_{\mathbf{x}}, \widehat{\mathbf{A}}_{\mathbf{x}}^{\mathsf{eval}}, \mathsf{ct}_{\mathbf{x}}^{\mathsf{eval}})$, where $\tilde{r}_{\mathbf{x}} = \mathsf{CPRF.ConstrainEval}(\widetilde{\mathsf{k}}_f^{\mathsf{con}}, \mathbf{x})$. However, the secret key $\mathbf{e}_{\mathbf{x}}$ satisfying $\mathbf{A}_{\mathbf{x}, r_{\mathbf{x}}}^{\mathsf{eq}} \mathbf{e}_{\mathbf{x}} = \mathbf{u}$ is useless for decryption because the (intermediate) matrices $\mathbf{A}_{\mathbf{x}}^{\mathsf{eval}}$ and $\widehat{\mathbf{A}}_{\mathbf{x}}^{\mathsf{eval}}$ in the single-framed box and the shadowed single-framed box, respectively, are different. Therefore, the tagging via CPRFs idea even fails to provide a correct ABE.

The main idea of Tsabary [46] to overcome this issue was taking advantage of the particular composition property of the lattice evaluation algorithms [10]. Specifically, for any matrix $\mathbf{A}$ and circuits h, g_1, and g_2, where h and $g_2 \circ g_1$ are described identically as circuits, the following evaluated matrices $\mathbf{A}_h$ and $\mathbf{A}_{g_2 \circ g_1}$ are the same, that is, $\mathbf{A}_h = \mathbf{A}_{g_2 \circ g_1}$:

1. $\mathbf{A}_h \leftarrow \mathsf{PubEval}(h, \mathbf{A})$,
2. $\mathbf{A}_{g_2 \circ g_1} \leftarrow \mathsf{PubEval}(g_2, \mathsf{PubEval}(g_1, \mathbf{A}))$.

Then, due to the correctness of $\mathsf{PubEval}$ and CtEval, when $\mathsf{ct} = \mathbf{s}^\top(\mathbf{A} - \mathbf{z} \otimes \mathbf{G}) + \mathsf{noise}$, ciphertexts ct_h and $\mathsf{ct}_{g_2 \circ g_1}$ are both of the form $\mathbf{s}^\top(\mathbf{A}_h - h(\mathbf{z}) \otimes \mathbf{G}) + \mathsf{noise}$. To take advantage of this property in the above CPRF idea, Tsabary required that the following algorithms are represented as identical circuits in case $f(\mathbf{x}) = 1$:

$$\mathsf{CPRF.Eval}(\cdot, \mathbf{x}) \equiv_{\mathsf{cir}} \mathsf{CPRF.ConstrainEval}(\mathsf{CPRF.Constrain}(\cdot, f), \mathbf{x}), \tag{9}$$

where $C \equiv_{\mathsf{cir}} C'$ denotes that circuits C and C' are identical.[4] Here, this corresponds to setting $h = \mathsf{CPRF.Eval}(\cdot, \mathbf{x})$, $g_1 = \mathsf{CPRF.Constrain}(\cdot, f)$, $g_2 = \mathsf{CPRF.ConstrainEval}(\cdot, \mathbf{x})$, and $\mathbf{z} = \widetilde{\mathsf{k}}^\top$ in the above. Tsabary [46] coins CPRFs with such a property as *conforming* CPRFs. Effectively, matrices $\mathbf{A}_{\mathbf{x}}^{\mathsf{eval}}$ and $\widehat{\mathbf{A}}_{\mathbf{x}}^{\mathsf{eval}}$ in Equations (6) and (8) are identical if we use such a conforming CPRF. Consequently, we have $\mathbf{A}_{\mathbf{x},r_{\mathbf{x}}}^{\mathsf{eq}} = \widehat{\mathbf{A}}_{\mathbf{x},r_{\mathbf{x}}}^{\mathsf{eq}}$. Therefore, decryption is now well-defined since the short vector $\mathbf{e}_{\mathbf{x}}$ can be used as expected.

The security proof of the scheme follows almost identically to the adaptive IBE setting: During the simulation, we first erase the information on $\widetilde{\mathsf{k}}$ from the challenge ciphertext and then apply adaptive pseudorandomness to replace $\widetilde{\mathsf{k}}_f^{\mathsf{con}}$ with the real constrained key $\mathsf{k}_f^{\mathsf{con}}$. Then, we undo the change and encode k in the challenge ciphertext in place of $\widetilde{\mathsf{k}}$. At this point, the reduction algorithm can embed its LWE problem in the challenge ciphertext. Note that we can swap $\widetilde{\mathsf{k}}_f^{\mathsf{con}}$ with $\mathsf{k}_f^{\mathsf{con}}$ because the ABE adversary can only obtain secret keys (that includes the output of $\mathsf{CPRF.Eval}(\mathsf{k}, \cdot)$) for attributes $\mathbf{x}$ such that $f(\mathbf{x}) = 0$. In particular, the adversary cannot use k_f to check whether the random tag associated with the secret key is generated by k or not.

The final remaining issue is whether such an adaptively secure conforming CPRF exists or not. Fortunately, the CPRF for bit-fixing predicates by Davidson et al. [23] (with a minor tweak) enjoyed such properties. Tsabary [46] further extended this CPRF to predicates expressed by t-CNF. Therefore, combining everything together, Tsabary obtained an adaptively secure (CP-)ABE for t-CNF policies.

Our Results. We are now prepared to explain our result. We first show why and how to weaken the conforming CPRF property required in the (semi-)generic

[4] More precisely, Tsabary [46] required that the circuit representation of $\mathsf{CPRF.Eval}(\cdot, \mathbf{x})$ and the effective *sub-circuit* of $\mathsf{CPRF.ConstrainEval}(\mathsf{CPRF.Constrain}(\cdot, f), \mathbf{x})$ are required to be the same.

construction of Tsabary [46]. We then present how to obtain such a CPRF for inner-products over $\mathbb{Z}$ from LWE building on top of the recent CPRF proposal of Davidson et al. [23]. By carefully combining them, we obtain the first lattice-based IPE over $\mathbb{Z}$. Finally, we briefly mention how to extend our IPE over $\mathbb{Z}$ to other types of useful ABE.

Weakening the Condition on Conforming CPRF. Combining the discussion thus far, an adaptively secure conforming CPRF for a more expressive constraint class $\mathcal{F}$ will immediately yield a (CP-)ABE for the policy class $\mathcal{F}$ based on Tsabary's proof methodology. Put differently, the goal now is to construct an adaptively secure CPRF such that for all $f \in \mathcal{F}$ and $\mathbf{x}$ where $f(\mathbf{x}) = 1$, Equation (9) holds. However, this turns out to be an extremely strong requirement which we only know how to construct using the CPRF for t-CNF [23,46]. This CPRF for t-CNF is based on a combinatoric approach using PRFs and differs significantly from all other (selectively secure) CPRFs for more expressive constraints that rely on algebraic tools such as bilinear-maps or lattices, e.g., [7,15,16,18,21,41]. That being said, there is one recent lattice-based CPRF for inner-products over $\mathbb{Z}$ by Davidson et al. [23] that comes somewhat close to what is required. Let us review their CPRF and explain how it fails short to fit in Tsabary's proof methodology.

A CPRF for inner-products over $\mathbb{Z}$ is a CPRF where the inputs and constraints are provided by vectors $\mathbf{x}, \mathbf{y} \in [-B, B]^\ell$ for some integer B. A constrained key $\mathsf{k}_{\mathbf{y}}^{\mathsf{con}}$ for vector $\mathbf{y}$ should allow to compute the same random value as the secret key k (i.e., the "base" seed) for all inputs $\mathbf{x}$ such that $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ over $\mathbb{Z}$. In Davidson et al. [23] the secret key k is simply a random matrix-vector pair $(\mathbf{S}, \mathbf{d})$ sampled uniformly random over $[-\bar{\beta}, \bar{\beta}]^{n \times \ell} \times [-\beta, \beta]^n$ for some integers $\bar{\beta}$ and β, where $\bar{\beta}$ is sub-exponentially large.[5] In addition, a matrix $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_{q'}^{n \times m}$ is provided as a public parameter. To evaluate on $\mathbf{x}$ using the secret key k, the $\mathsf{CPRF.Eval}$ algorithm first converts $\mathbf{B}$ to a specific matrix $\mathbf{B}_{\mathbf{x}}$ associated to $\mathbf{x}$ (whose detail is irrelevant for this overview). Then, it computes a vector $\mathsf{k}_{\mathbf{x}}^{\mathsf{int}} := \mathbf{S}\mathbf{x} \in \mathbb{Z}^n$ called an *intermediate key*, and finally outputs the random value $r_{\mathbf{x}} = \lfloor \mathsf{k}_{\mathbf{x}}^{\mathsf{int}\top} \mathbf{B}_{\mathbf{x}} \rceil_p \in \mathbb{Z}_p^m$. Here, $\lfloor a \rceil_p$ denotes rounding of an element $a \in \mathbb{Z}_{q'}$ to $\mathbb{Z}_p$ by multiplying it by (p/q') and rounding the result.[6] The constrained key $\mathsf{k}_{\mathbf{y}}^{\mathsf{con}}$ is simply defined as $\mathsf{k}_{\mathbf{y}}^{\mathsf{con}} := \mathbf{S} + \mathbf{d} \otimes \mathbf{y}^\top \in \mathbb{Z}^{n \times \ell}$. To evaluate on $\mathbf{x}$ using the constrained key $\mathsf{k}_{\mathbf{y}}^{\mathsf{con}}$, the $\mathsf{CPRF.ConstrainEval}$ algorithm first prepares $\mathbf{B}_{\mathbf{x}}$ as done by $\mathsf{CPRF.Eval}$ and then computes the *constrained* intermediate key $\mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}} := (\mathbf{S} + \mathbf{d} \otimes \mathbf{y}^\top)\mathbf{x} \in \mathbb{Z}^{n \times \ell}$, and finally outputs the random value $r'_{\mathbf{x}} = \lfloor \mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}\top} \mathbf{B}_{\mathbf{x}} \rceil_p \in \mathbb{Z}_p^m$. Observe that if $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ over $\mathbb{Z}$, then $\mathsf{k}_{\mathbf{x}}^{\mathsf{int}} = \mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}}$. Therefore, $\mathsf{CPRF.Eval}(\mathsf{k}, \mathbf{x}) = \mathsf{CPRF.ConstrainEval}(\mathsf{k}_{\mathbf{y}}, \mathbf{x})$ in case $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ as desired. Davidson et al. [23] proved that such a CPRF is adaptively secure based on the LWE assumption with sub-exponential modulus size.

[5] In their original scheme, $\mathbf{d}$ is not included in the secret key but generated when constraining the secret key. However, this modification is w.l.o.g and will be vital for our purpose.

[6] Looking ahead, we note the moduli (q', p) used by the CPRF is different from the modulus q used by the ABE.

On first glance this CPRF may seem to satisfy the conforming property (Equation (9)) since the secret key $\mathsf{k} = \mathbf{S}$ and the constrained key $\mathsf{k}_{\mathbf{y}}^{\mathsf{con}} = \mathbf{S} + \mathbf{d} \otimes \mathbf{y}^\top$ are both matrices over $\mathbb{Z}^{n \times \ell}$, and the intermediate keys $\mathsf{k}_{\mathbf{x}}^{\mathsf{int}}$ and $\mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}}$ are equivalent in case $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ and are used identically (as a circuit) to compute $r_{\mathbf{x}}$. However, under closer inspection, it is clear that Equation (9) does not hold. Specifically, $\mathsf{CPRF.Constrain}(\mathsf{k}, \mathbf{y})$ computes $\mathsf{k}_{\mathbf{y}}^{\mathsf{con}} = (\mathbf{S} + \mathbf{d} \otimes \mathbf{y}^\top)$; a computation that depends on the constraint vector $\mathbf{y}$, while $\mathsf{CPRF.Eval}(\mathsf{k}, \mathbf{x})$ does not internally perform such computation. Therefore, $\mathsf{CPRF.Eval}(\cdot, \mathbf{x})$ cannot be identical as a circuit as $\mathsf{CPRF.ConstrainEval}(\mathsf{CPRF.Constrain}(\cdot, \mathbf{y}), \mathbf{x})$. In the context of ABE, this means that the KeyGen algorithm and Enc/Dec algorithms will not be able to agree on the same matrix, and hence, correctness no longer holds. Although both algorithms CPRF.Eval and CPRF.ConstrainEval share a striking resemblance, it seems one step short of satisfying the conforming property of Tsabary.

Our main idea to overcome this issue is weakening the conforming property required by Tsabary [46] by noticing another particular *linearity* property of the lattice evaluation algorithms of [10]. Specifically, for any matrix $\mathbf{A}$ and *linear functions* h, g_1, and g_2 such that h and $g_2 \circ g_1$ are *functionally equivalent*, the matices $\mathbf{A}_h$ and $\mathbf{A}_{g_2 \circ g_1}$ evaluated using PubEval as in Items 1 and 2 are in fact equivalent (i.e., $\mathbf{A}_h = \mathbf{A}_{g_2 \circ g_1}$). By correctness of PubEval and CtEval, we then also have $\mathsf{ct}_h = \mathsf{ct}_{g_2 \circ g_1}$. Here, the main observation is that we no longer require the strong property of $h \equiv_{\mathsf{cir}} g_2 \circ g_1$, but only require a slightly milder property of h and $g_2 \circ g_1$ being functionally equivalent, that is, have the same input/output.

Let us see how this property can be used. Notice that the above CPRF of Davidson et al. [23] has the following structure. Algorithm $\mathsf{CPRF.Eval}(\mathsf{k}, \mathbf{x})$ can be broken up in linear and non-linear algorithms: $\mathsf{CPRF.EvalLin}(\mathsf{k}, \mathbf{x}) \to \mathsf{k}_{\mathbf{x}}^{\mathsf{int}}$ and $\mathsf{CPRF.EvalNonLin}(\mathsf{k}_{\mathbf{x}}^{\mathsf{int}}, \mathbf{x}) \to r_{\mathbf{x}}$.[7] Namely, we have

$$\mathsf{CPRF.Eval}(\mathsf{k}, \mathbf{x}) = \mathsf{CPRF.EvalNonLin}(\mathsf{CPRF.EvalLin}(\mathsf{k}, \mathbf{x}), \mathbf{x}).$$

Similarly, $\mathsf{CPRF.ConstrainEval}(\mathsf{k}_{\mathbf{y}}, \mathbf{x})$ can be broken up in linear and non-linear algorithms: $\mathsf{CPRF.ConstrainEvalLin}(\mathsf{k}_{\mathbf{y}}^{\mathsf{con}}, \mathbf{x}) \to \mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}}$ and $\mathsf{CPRF.ConstrainEvalNonLin}(\mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}}, \mathbf{x}) \to r_{\mathbf{x}}$. In addition, from above, we know that we have the following property:

1. if $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ over $\mathbb{Z}$, then $\mathsf{CPRF.EvalLin}(\cdot, \mathbf{x})$ and $\mathsf{CPRF.ConstrainEvalLin}(\mathsf{CPRF.Constrain}(\cdot, \mathbf{y}), \mathbf{x})$ are both *linear* functions that are *functionally equivalent* (in particular, $\mathsf{k}_{\mathbf{x}}^{\mathsf{int}} = \mathsf{k}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}}$), and
2. the non-linear algorithms satisfy $\mathsf{CPRF.EvalNonLin}(\cdot, \mathbf{x}) \equiv_{\mathsf{cir}} \mathsf{CPRF.ConstrainEvalNonLin}(\cdot, \mathbf{x})$. Namely, they are *identical circuits*.

Importing these properties to the ABE setting, we get a transition of matrices and ciphertext for KeyGen, Enc, and Dec as in Figure 5.

Notice the matrices in red ($\mathbf{A}_{\mathbf{x}}^{\mathsf{int}}$ and $\mathbf{A}_{\mathbf{y},\mathbf{x}}^{\mathsf{con\text{-}int}}$) are identical due to the property in Item 1 and the linearity property of PubEval and CtEval. Moreover, due to

[7] Concretely, the non-linear part does a rounding operation modulo a certain integer p followed by an evaluation of a hash function.

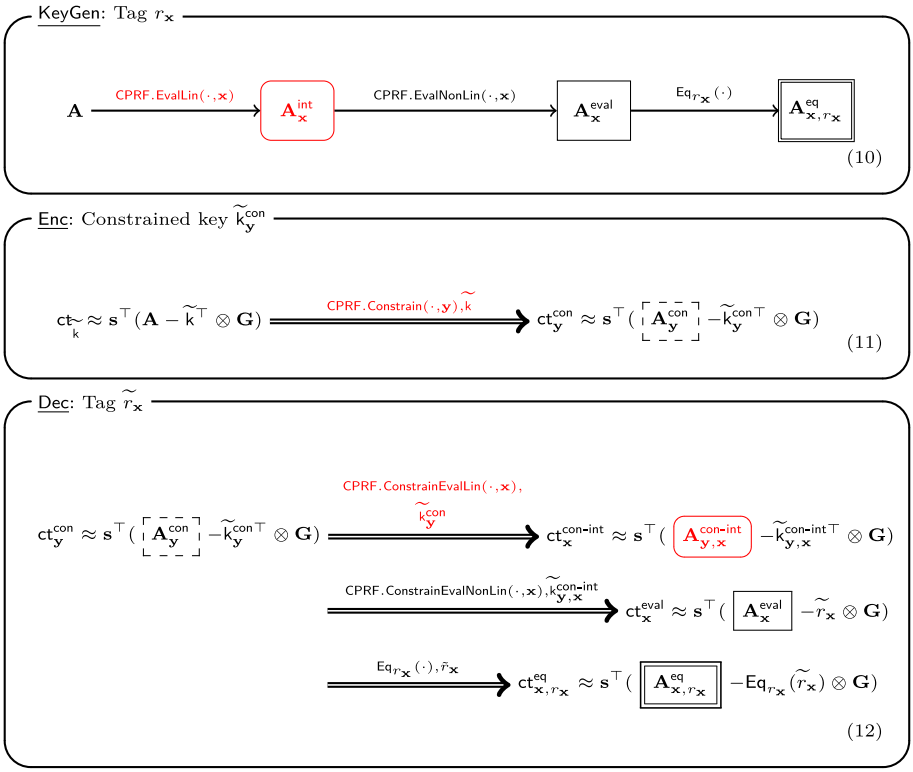

Fig. 5. Illustration of our adaptively secure IPE.

the property in Item 2, the subsequent evaluated ciphertexts $\mathsf{ct}^{\mathsf{eval}}_{\mathbf{x}}$ and $\mathsf{ct}^{\mathsf{eq}}_{\mathbf{x},r_{\mathbf{x}}}$ correctly encode the matrices $\mathbf{A}^{\mathsf{eval}}_{\mathbf{x}}$ and $\mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r_{\mathbf{x}}}$, respectively, which correspond to those computed during KeyGen. Combining all of these observations, it seems we have successfully weakened the conforming property required by Tsabary [46] and showed that the CPRF of Davidson et al. [23] suffices to instantiate the generic (CP-)ABE construction. However, we show that a problem still remains.

Bit Decomposing and Tweaking Davidson et al.'s CPRF [23]. To understand the problem, let us take a closer look at how the CtEval algorithm is used in Equations (11) and (12). First, observe that the output of the linear function $\mathsf{CPRF.EvalLin}(\mathsf{k},\mathbf{x})$, or equivalently, the output of $\mathsf{CPRF.ConstrainEvalLin}(\mathsf{CPRF.Constrain}(\mathsf{k},\mathbf{y}),\mathbf{x})$ is over $\mathbb{Z}$ rather than over $\{0,1\}$. More specifically, the output $\mathsf{k}^{\mathsf{int}}_{\mathbf{x}} (= \mathsf{k}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}})$ is of the form $\mathbf{Sx} \in [-\tilde{\beta},\tilde{\beta}]^n$, where $\tilde{\beta}$ is some *sub-exponentially* large integer. Therefore, the ciphertext $\mathsf{ct}^{\mathsf{con\text{-}int}}_{\mathbf{x}} \approx \mathbf{s}^\top(\mathbf{A}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}} - \widetilde{\mathsf{k}}^{\mathsf{con\text{-}int}\top}_{\mathbf{y},\mathbf{x}} \otimes \mathbf{G})$ computed within the Dec algorithm encodes $\widetilde{\mathsf{k}}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}}$ as *integers* over $[-\tilde{\beta},\tilde{\beta}]^n$. Now, the Dec algorithm must further convert this ciphertext to $\mathsf{ct}^{\mathsf{eval}}_{\mathbf{x}} \approx \mathbf{s}^\top(\mathbf{A}^{\mathsf{eval}}_{\mathbf{x}} - \tilde{r}_{\mathbf{x}} \otimes \mathbf{G})$, where $\tilde{r}_{\mathbf{x}} = \mathsf{CPRF.ConstrainEvalNonLin}(\mathsf{k}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}},\mathbf{x}) = \lfloor \mathsf{k}^{\mathsf{con\text{-}int}\top}_{\mathbf{x},\mathbf{y}} \mathbf{B}_{\mathbf{x}} \rceil_p \in \mathbb{Z}_p^m$. The problem is: is this efficiently computable? Since $\mathbf{B}_{\mathbf{x}}$ can be precomputed

and $\mathsf{k}^{\mathsf{con\text{-}int}\top}_{\mathbf{x},\mathbf{y}}\mathbf{B}_{\mathbf{x}}$ is a linear function of $\mathsf{k}^{\mathsf{con\text{-}int}}_{\mathbf{x},\mathbf{y}}$, the problem boils down to the following question:

> *Given* $x \in [-\tilde{\beta}, \tilde{\beta}]$ *and* $\mathsf{ct} = \mathbf{s}^\top(\mathbf{A} + x \otimes \mathbf{G}) + \mathsf{noise} \pmod q$ *as inputs, can we efficiently compute* $\mathsf{ct}_p \approx \mathbf{s}^\top(\mathbf{A}_p + \lfloor x \rfloor_p \cdot \mathbf{G})$, *where* $0 < \tilde{\beta} < p < q$ *and* $\tilde{\beta}$ *is sub-exponentially large and* $\mathbf{A}_p$ *is some publicly computable matrix independent of the value* x*?*

Unfortunately, this problem turns out to be quite difficult, and as far as our knowledge goes, we do not know how to achieve this.[8] One of the main reason for the difficulty is that we cannot efficiently simulate arithmetic operations over the ring $\mathbb{Z}_p$ by an arithmetic circuit over another ring $\mathbb{Z}_q$ when the input is provided as a sub-exponentially large integer (and not as a bit-string).

To circumvent this seemingly difficult problem, we incorporate two additional ideas. First, we consider an easier problem compared to above where $\tilde{\beta}$ is guaranteed to be only *polynomially* large. In this case, we show that the problem is indeed solvable. Notably, if $|x|$ is only polynomially large, then we can efficiently compute the bit-decomposition of x by an arithmetic circuit over the ring $\mathbb{Z}_q$ by using Lagrange interpolation. That is, there exists an efficiently computable degree-$2\tilde{\beta}$ polynomial p_i over $\mathbb{Z}_q$ such that $p_i(x)$ computes the i-th bit of the bit-decomposition of x. Therefore, given $\mathsf{ct} \approx \mathbf{s}^\top(\mathbf{A} + x \otimes \mathbf{G})$ as input, we first compute $\mathsf{ct}_{\mathsf{bd}} \approx \mathbf{s}^\top(\mathbf{A}_{\mathsf{bd}} + \mathsf{BitDecomp}(x) \otimes \mathbf{G})$ by using the polynomials $(p_i)_i$, where $\mathbf{A}_{\mathsf{bd}} = \mathsf{PubEval}(\mathbf{A}, \mathsf{BitDecomp}(\cdot))$. We then compute $\mathsf{ct}_p \approx \mathbf{s}^\top(\mathbf{A}_p + \lfloor x \rfloor_p \otimes \mathbf{G})$, where we use the fact that arithmetic operations over the ring $\mathbb{Z}_p$ can be efficiently simulated with an arithmetic circuit over another ring $\mathbb{Z}_q$ in case the input is provided as a bit-string.

The remaining problem is whether $\tilde{\beta}$ in the CPRF of Davidson et al. [23] can be set to be polynomially large rather than sub-exponentially large. Very roughly, Davidson et al. required $\tilde{\beta}$ to be sub-exponentially large to argue that with all but a negligible probability, the absolute value of *all the entries* in $\mathbf{S} \in \mathbb{Z}^{n \times \ell}$ is smaller than some specified value. However, we notice that we can complete the same security proof by only requiring that the absolute value of *most of the entries* in $\mathbf{S}$ is smaller than a specified value. This small change allows us to use a finer probabilistic argument on the entries of $\mathbf{S}$, which in return, allows us to set $\tilde{\beta}$ only polynomially large.

By combining all the pieces, we obtain the first lattice-based adaptively secure IPE over $\mathbb{Z}$ with polynomial-sized entries. We note that our construction requires LWE with a sub-exponential modulus since the underlying CPRF of [23] requires it, and also, since we need to homomorphically compute the non-linear circuit $\mathsf{CPRF.EvalNonLin}$.

Extending IPE Over $\mathbb{Z}$ to Other ABEs. Finally, we also show how to extend our adaptively secure IPE over $\mathbb{Z}$ with polynomial-sized entries to other useful ABE using generic conversions. That is, the ideas are not limited to our specific lattice-based construction. Specifically, we obtain the following three lattice-based adaptively secure ABEs for the first time: IPE over the ring $\mathbb{Z}_p$ for $p =$

[8] We note that a solution to this question will directly give us the desired result.

$\mathsf{poly}(\kappa)$, fuzzy IBE for small and large universes with threshold T. The first two generic conversions are almost folklore. To obtain fuzzy IBE for large universe, we use error correcting codes with a polynomial-sized alphabet (such as Reed-Solomon codes [42]) to encode an exponentially large element to a string of polynomially large elements with polynomial length. We then use the fuzzy IBE for small universe with an appropriate threshold to simulate the large universe.

1.3 Related Works

Brakerski and Vaikuntanathan [17] constructed a lattice-based ABE for all circuits with a weaker adaptive security called the *semi-adaptive security*, where an adversary can declare the challenge attribute after seeing the public parameter but before making any key query. Subsequently, Goyal, Koppula and Waters [30] showed that we can convert any selectively secure ABE into a semi-adaptively secure one.

Recently, Wang et al. [47] gave a framework to construct lattice-based adaptively secure ABE by extending the dual system framework [48] into the lattice setting. However, their instantiation based on the LWE assumption only yields *bounded collusion-resistant* ABE where an adversary can obtain only bounded number of decryption keys that is fixed at the setup phase. We note that such an ABE trivially follows from the bounded collusion-resistant functional encryption scheme based on any PKE by Gorbunov, Vaikuntanathan, and Wee [27].

2 Preliminaries

We use standard cryptographic notations and refer the readers to the full version for reference.

2.1 Lattices

In this work, we only use standard tools from lattices such as bounding norms of discrete Gaussian distributions, gadget matrices, and sampling with trapdoors. Therefore, we omit the details to the full version. Below, we introduce the main hardness assumption we use in this work for completeness.

Definition 2.1 ([43], Learning with Errors). *For integers n, m, a prime $q > 2$, an error distribution χ over $\mathbb{Z}$, and a PPT algorithm $\mathcal{A}$, the advantage for the* learning with errors problem $\mathsf{LWE}_{n,m,q,\chi}$ *of $\mathcal{A}$ is defined as follows:*

$$\mathsf{Adv}^{\mathsf{LWE}_{n,m,q,\chi}}_{\mathcal{A}} = \left| \Pr\left[\mathcal{A}\left(\mathbf{A}, \mathbf{s}^\top \mathbf{A} + \mathbf{z}^\top\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathbf{A}, \mathbf{b}^\top\right) = 1\right] \right|$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{b} \leftarrow \mathbb{Z}_q^m$, $\mathbf{z} \leftarrow \chi^m$. We say that the LWE *assumption holds if $\mathsf{Adv}^{\mathsf{LWE}_{n,m,q,\chi}}_{\mathcal{A}}$ is negligible for all PPT algorithm $\mathcal{A}$.*

The (decisional) $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$ for $\alpha q > 2\sqrt{n}$ has been shown by Regev [43] via a quantum reduction to be as hard as approximating the worst-case SIVP and GapSVP problems to within $\tilde{O}(n/\alpha)$ factors in the ℓ_2-norm in the worst case. In the subsequent works (partial) dequantumization of the reduction were achieved [14,40]. The worst-case problems are believed to be hard even for subexponential approximation factors, and in particular, the LWE problem with subexponential modulus size is believed to be hard. We note that this is different from assuming the subexponential LWE assumption where we allow for adversaries even with subexponentially small advantage.

2.2 Attribute-Based Encryption

Let $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ where $\mathcal{X}$ and $\mathcal{Y}$ are sets. An attribute-based encryption (ABE) for P (with the message space $\{0,1\}$) consists of PPT algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$: $\mathsf{Setup}(1^\kappa)$ outputs a pair public parameter and master secret key $(\mathsf{pp}, \mathsf{msk})$; $\mathsf{KeyGen}(\mathsf{pp}, \mathsf{msk}, x)$ outputs a secret key sk_x for attribute x; $\mathsf{Enc}(\mathsf{pp}, y, M)$ outputs a ciphertext ct_y for attribute y and message $\mathsf{M} \in \{0,1\}$; and $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_x, \mathsf{ct}_y)$ outputs M if $\mathsf{P}(x,y) = 1$.

An ABE is said to be (adaptively) payload-hiding, if it is infeasible for an adversary to tell apart a random ciphertext and a valid ciphertext for attribute y^* and message M^* of its choice, even if it is given polynomially many secret keys sk_x for $\mathsf{P}(x,y) = 0$. Here, adaptive security dictates that an adversary can adaptively choose the challenge attribute y^* even after seeing polynomially many secret keys sk_x. The formal definition is omitted to the full version.

Inner-product Encryption. In this study, we consider ABEs for the following predicate. Let P be the inner-product predicate with domain $\mathcal{X} = \mathcal{Y} = \mathcal{Z}^n$ where $\mathcal{Z}$ is a subset of $\mathbb{Z}$. That is, for $\mathbf{x}, \mathbf{y} \in \mathcal{Z}^n$, $\mathsf{P}(\mathbf{x}, \mathbf{y}) = 1$ if $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ and $\mathsf{P}(\mathbf{x}, \mathbf{y}) = 0$ otherwise. We call this inner-production encryption (IPE) *over the integers* ($\mathbb{Z}$).

We also consider a variant where the inner-product is taken over $\mathbb{Z}_p$ for p a prime. Concretely, let $\mathsf{P}_{\mathsf{mod}}$ be the inner-product predicate with domain $\mathcal{X} = \mathcal{Y} = \mathbb{Z}_p^n$ such that for $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_p^n$, $\mathsf{P}_{\mathsf{mod}}(\mathbf{x}, \mathbf{y}) = 1$ if $\langle \mathbf{x}, \mathbf{y} \rangle = 0 \mod p$ and $\mathsf{P}(\mathbf{x}, \mathbf{y}) = 0$ otherwise. We call this IPE *over* $\mathbb{Z}_p$.

Fuzzy Identity-based Encryption. We also consider the following predicate. Let $\mathsf{P}_{\mathsf{fuz}}$ be the fuzzy predicate with domain $\mathcal{X}^n = \mathcal{Y}^n = \mathcal{D}^n$ and *threshold* $T(> 0)$ such that for $\mathbf{x}, \mathbf{y} \in \mathcal{D}^n$, $\mathsf{P}_{\mathsf{fuz}}(\mathbf{x}, \mathbf{y}) = 1$ if $\mathsf{HD}(\mathbf{x}, \mathbf{y}) \leq n - T$ and $\mathsf{P}_{\mathsf{fuz}}(\mathbf{x}, \mathbf{y}) = 0$ otherwise. Here, $\mathsf{HD} : \mathcal{D}^n \times \mathcal{D}^n \to [0, n]$ denotes the hamming distance. That is, if $\mathbf{x}$ and $\mathbf{y}$ are identical in more than T-positions, then $\mathsf{P}_{\mathsf{fuz}}(\mathbf{x}, \mathbf{y}) = 1$. We call this fuzzy identity-based encryption (IBE) for *small universe* when $|\mathcal{D}| = \mathsf{poly}(\kappa)$, and fuzzy IBE for *large universe* when $|\mathcal{D}| = \exp(\kappa)$.

2.3 Constrained Pseudorandom Functions

A constrained pseudorandom function for $(\mathcal{D}, \mathcal{R}, \mathcal{K}, \mathcal{C})$ is defined by the five PPT algorithms $\Pi_{\mathsf{CPRF}} = (\mathsf{CPRF.Setup}, \mathsf{CPRF.Gen}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain},$

$\mathsf{CPRF.ConstrainEval}$) where: $\mathsf{CPRF.Setup}(1^\kappa)$ outputs a set of public parameter pp; $\mathsf{CPRF.Gen}(\mathsf{pp})$ outputs a master key K; $\mathsf{CPRF.Eval}(\mathsf{pp}, \mathsf{K}, x)$ outputs a random value r; $\mathsf{CPRF.Constrain}(\mathsf{K}, C)$ outputs a constrained key $\mathsf{K}_C^{\mathsf{con}}$ associated with constraint C; and $\mathsf{CPRF.ConstrainEval}(\mathsf{pp}, \mathsf{K}_C^{\mathsf{con}}, x)$ outputs the same value as $\mathsf{CPRF.Eval}(\mathsf{pp}, \mathsf{K}, x)$ when $C(x) = 1$.

A CPRF is said to be (adaptive) *pseudorandomness on constrained points*, when informally, it infeasible for an adversary to evaluate on a point when only given constrained keys that are constrained on that particular point. Here, adaptive security dictates that an adversary can adaptively query the constrained keys even after seeing polynomially many evaluations. The formal definition is omitted to the full version.

3 Lattice Evaluations

In this section, we show various lattice evaluation algorithms that will be used in the description of our IPE scheme in Sec. 5. We start by recalling the following lemma, which is an abstraction of the evaluation algorithms developed in a long sequence of works [10,26,29,37].

Lemma 3.1 ([46, Theorem 2.5]). *There exist efficient deterministic algorithms* EvalF *and* EvalFX *such that for all* $n, q, \ell \in \mathbb{N}$ *and* $m \geq n\lceil \log q \rceil$*, for any depth* d *boolean circuit* $f : \{0,1\}^\ell \to \{0,1\}^k$*, input* $x \in \{0,1\}^\ell$*, and matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times m(\ell+1)}$*, the outputs* $\mathbf{H} := \mathsf{EvalF}(f, \mathbf{A})$ *and* $\widehat{\mathbf{H}} := \mathsf{EvalFX}(f, x, \mathbf{A})$ *are both in* $\mathbb{Z}^{m(\ell+1) \times m(k+1)}$ *and it holds that* $\|\mathbf{H}\|_\infty, \|\widehat{\mathbf{H}}\|_\infty \leq (2m)^d$*, and*

$$[\mathbf{A} - (1, x) \otimes \mathbf{G}]\widehat{\mathbf{H}} = \mathbf{A}\mathbf{H} - (1, f(x)) \otimes \mathbf{G} \mod q.$$

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

Here, we note that unlike in the original theorem [46, Theorem 2.5], we would require the constant 1 term to handle functions with a constant term. More details are provided in the full version.

In the following, we generalize the above lemma so that we can treat the case where x and $f(x)$ are integer vectors rather than bit strings. We first consider the case where function f is a linear function over $\mathbb{Z}^\ell$ in Sec. 3.1. The algorithm we give is essentially the same as that given in the previous work [10], but we will make a key observation that the evaluation results of two functions are the same as long as they are functionally equivalent *even if they are expressed as different (arithmetic) circuits*. In Sec. 3.2, we consider the case where f is a specific type of non-linear function taking a vector $\mathbf{x} \in \mathbb{Z}^\ell$ as input; f initially computes a binary representation of the input $\mathbf{x}$, and then computes an arbitrary function represented by a boolean circuit over that binarized input. We note that an evaluation algorithm for arithmetic circuits over $\mathbb{Z}$ in previous work [10] is not enough for our purpose. This is because the binary representation of an integer may not be efficiently computable by an arithmetic circuit over $\mathbb{Z}$ in case the integer is super-polynomially large.

3.1 Linear Evaluation

Here, we deal with linear functions over $\mathbb{Z}$ that are expressed by arithmetic circuits.

Definition 3.1. *For a (homogeneous) linear function $f : \mathbb{Z}^\ell \to \mathbb{Z}^k$, we denote the unique matrix that represents f by $\mathbf{M}_f$. That is, $\mathbf{M}_f = (m_{i,j})_{i\in[\ell],j\in[k]} \in \mathbb{Z}^{\ell\times k}$ is the matrix such that we have $f(\mathbf{x})^\top = \mathbf{x}^\top \cdot \mathbf{M}_f$. We denote $\|f\|_\infty$ to mean $\|\mathbf{M}_f\|_\infty$ and call $\|f\|_\infty$ the norm of f.*

The following lemma gives an evaluation algorithm for linear functions. The proof can be checked easily and is omitted to the full version.

Lemma 3.2. *There exist efficient deterministic algorithms* $\mathsf{EvalLin}$ *such that for all $n, m, q, \ell \in \mathbb{N}$, for any linear function $f : \mathbb{Z}^\ell \to \mathbb{Z}^k$, input $\mathbf{x} \in \mathbb{Z}^\ell$, and matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m(\ell+1)}$, the output $\overline{\mathbf{M}}_f := \mathsf{EvalLin}(f)$ is in $\mathbb{Z}^{m(\ell+1)\times m(k+1)}$ and it holds that $\|\overline{\mathbf{M}}_f\|_\infty = \max\{1, \|f\|_\infty\}$, and*

$$[\mathbf{A} - (1, \mathbf{x}^\top) \otimes \mathbf{G}]\overline{\mathbf{M}}_f = \mathbf{A}\overline{\mathbf{M}}_f - (1, f(\mathbf{x})^\top) \otimes \mathbf{G} \mod q.$$

Moreover, for any tuple of linear functions $f : \mathbb{Z}^\ell \to \mathbb{Z}^k$, $g : \mathbb{Z}^k \to \mathbb{Z}^t$, and $h : \mathbb{Z}^\ell \to \mathbb{Z}^t$ such that $g \circ f(\mathbf{x}) = h(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{Z}^\ell$, the outputs $\overline{\mathbf{M}}_f := \mathsf{EvalLin}(f)$, $\overline{\mathbf{M}}_g := \mathsf{EvalLin}(g)$ and $\overline{\mathbf{M}}_h := \mathsf{EvalLin}(h)$ satisfy $\overline{\mathbf{M}}_f\overline{\mathbf{M}}_g = \overline{\mathbf{M}}_h$.

Looking ahead, the latter part of the above lemma is a key property for our generalization of the Tsabary's framework [46] when constructing adaptively secure ABE. Note that in the general non-linear case, an analogue of this property only holds when $g \circ f$ and h are expressed exactly as the same circuit (See Lemma 3.1).

3.2 Non-linear Evaluation

Next, we consider the non-linear case where f takes as input a vector $\mathbf{x} \in \mathbb{Z}^\ell$. Specifically, f first computes the binary decomposition of $\mathbf{x}$, and then performs an arbitrary computation represented by a boolean circuit. Since the latter part of the computation can be handled by Lemma 3.1, all we have to do is to give a homomorphic evaluation algorithm that handles the former part of the computation. The following lemma enables us to do this as long as $\|\mathbf{x}\|_\infty$ is bounded by some polynomial in κ. At a high level, when $\|\mathbf{x}\|_\infty$ is only a polynomial, we would be able to efficiently compute the bit-decomposition of $\mathbf{x}$ using Lagrange interpolation. We omit the proof to the full version. In the statement below, we focus on the case of $\ell = 1$.

Lemma 3.3. *There exist efficient deterministic algorithms* EvalBD *and* $\mathsf{EvalBDX}$ *such that for all $n, m, M \in \mathbb{N}$, prime q satisfying $q > 2M + 1$ and $m \geq n\lceil \log q \rceil$, $x \in [-M, M]$, and for any matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times 2m}$, the outputs*

$\mathbf{H} := \mathsf{EvalBD}(1^M, \mathbf{A})$ *and* $\widehat{\mathbf{H}} := \mathsf{EvalBDX}(1^M, x, \mathbf{A})$ *are both in* $\mathbb{Z}^{2m \times m\lceil \log q \rceil}$ *and it holds that* $\|\mathbf{H}\|_\infty, \|\widehat{\mathbf{H}}\|_\infty \leq (2mM)^{2M+1}$*, and*

$$[\mathbf{A} - (1, x) \otimes \mathbf{G}]\widehat{\mathbf{H}} = \mathbf{AH} - \mathsf{BitDecomp}(x) \otimes \mathbf{G} \mod q \tag{13}$$

where $\mathsf{BitDecomp}(x) \in \{0,1\}^{\lceil \log q \rceil}$ *denotes the bit decomposition of* x*.*

Finally, we combine Lemmata 3.1 and 3.3, to obtain our desired lemma. Let q and M be integers such that $q > 2M+1$. In the following lemma, we deal with function $f : [-M, M]^\ell \to \{0,1\}^k$ that can be represented by a Boolean circuit $\tilde{f} : \{0,1\}^{\ell\lceil \log q \rceil} \to \{0,1\}^k$ in the sense that we have

$$f(\mathbf{x}) = \tilde{f}(\mathsf{BitDecomp}(x_1), \ldots, \mathsf{BitDecomp}(x_\ell))$$

for any $\mathbf{x} \in [-M, M]^\ell$. The proof is quite standard and is omitted to the full version.

Lemma 3.4. *There exist efficient deterministic algorithms* $\mathsf{EvalF}^{\mathsf{bd}}$ *and* $\mathsf{EvalFX}^{\mathsf{bd}}$ *such that for all* $n, m, \ell, M \in \mathbb{N}$*, prime* q *satisfying* $q > 2M + 1$ *and* $m \geq n\lceil \log q \rceil$*, for any function* $f : [-M, M]^\ell \to \{0,1\}^k$ *that can be expressed as an efficient depth* d *boolean circuit* $\tilde{f} : \{0,1\}^{\ell\lceil \log q \rceil} \to \{0,1\}^k$*, for every* $\mathbf{x} \in [-M, M]^\ell$*, and for any matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times m(\ell+1)}$*, the outputs* $\mathbf{H} := \mathsf{EvalF}^{\mathsf{bd}}(1^M, f, \mathbf{A})$ *and* $\widehat{\mathbf{H}} := \mathsf{EvalFX}^{\mathsf{bd}}(1^M, f, \mathbf{x}, \mathbf{A})$ *are both in* $\mathbb{Z}^{m(\ell+1) \times m(k+1)}$ *and it holds that* $\|\mathbf{H}\|_\infty$*,* $\|\widehat{\mathbf{H}}\|_\infty \leq \ell\lceil \log q \rceil (2mM)^{d+2M+2}$ *and*

$$[\mathbf{A} - (1, \mathbf{x}^\top) \otimes \mathbf{G}]\widehat{\mathbf{H}} = \mathbf{AH} - (1, f(\mathbf{x})) \otimes \mathbf{G} \mod q. \tag{14}$$

4 IPE-Conforming CPRF

In this section, we introduce the notion of *IPE-conforming* CPRF and instantiate it from the LWE assumption. An IPE-conforming CPRF is the main building block for our adaptively secure IPE schemes. Although Tsabary presents how to achieve adaptively secure ABE by using *conforming* CPRFs, the requirements on conforming CPRFs are quite strong and it seems very difficult to achieve such conforming CPRFs for inner-products. To achieve adaptively secure IPE, we relax the requirements.

4.1 Definition

Here, we define an IPE-conforming CPRF.

Definition 4.1. *A CPRF scheme* $\Pi_{\mathsf{CPRF}} = (\mathsf{CPRF.Setup}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.ConstrainEval})$ *that supports inner products over* $\mathcal{D} := [-B, B]^\ell \subset \mathbb{Z}^\ell$ *is said to be IPE-conforming if it satisfies the following properties:*

– *Partial linear evaluation (Definition 4.2)*
– *Key simulation (Definition 4.3)*
– *Uniformity (Definition 4.4)*

The partial linear evaluation property is a relaxed variant of the gradual evaluation property for conforming CPRFs defined by Tsabary [46]. Recall that the gradual evaluation property of Tsabary [46] requires that (a sub-circuit of) the composition of CPRF.Constrain and CPRF.ConstrainEval is identical to CPRF.Eval *as a circuit*. On the other hand, we only require that they are identical as (arithmetic) circuits *excluding the linear computation*. The precise definition follows.

Definition 4.2 (Partial linear evaluation). *The algorithm* CPRF.Eval *(resp.* CPRF.ConstrainEval*) can be divided into a linear part* CPRF.EvalLin *(resp.* CPRF.ConstrainEvalLin*) and a non-linear part* CPRF.EvalNonLin *(resp.* CPRF.ConstrainEvalNonLin*) with the following syntax:*

– $\mathsf{CPRF.EvalLin}(\mathsf{K}, \mathbf{x}) \to \mathsf{K}^{\mathsf{int}}_{\mathbf{x}} \in \mathbb{Z}^{\xi}$,
– $\mathsf{CPRF.EvalNonLin}(\mathsf{pp}, \mathsf{K}^{\mathsf{int}}_{\mathbf{x}}, \mathbf{x}) \to \mathsf{PRF}(\mathsf{K}, \mathbf{x})$,
– $\mathsf{CPRF.ConstrainEvalLin}(\mathsf{K}^{\mathsf{con}}_{\mathbf{y}}, \mathbf{x}) \to \mathsf{K}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}} \in \mathbb{Z}^{\xi}$,
– $\mathsf{CPRF.ConstrainEvalNonLin}(\mathsf{pp}, \mathsf{K}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}}, \mathbf{x}) \to \mathsf{PRF}(\mathsf{K}, \mathbf{x})$,

where the superscript int *stands for "intermediate key" and* $\mathsf{K}^{\mathsf{con}}_{\mathbf{y}}$ *denotes the constrained key for the inner-product constraint for vector* $\mathbf{y}$*. Specifically, we have*

$$\mathsf{CPRF.EvalNonLin}(\mathsf{pp}, \mathsf{CPRF.EvalLin}(\mathsf{K}, \mathbf{x}), \mathbf{x}) = \mathsf{CPRF.Eval}(\mathsf{K}, \mathbf{x})$$

and

$$\mathsf{CPRF.ConstrainEvalNonLin}(\mathsf{pp}, \mathsf{CPRF.ConstrainEvalLin}(\mathsf{K}, \mathbf{x}), \mathbf{x}) = \mathsf{CPRF.ConstrainEval}(\mathsf{K}, \mathbf{x}).$$

We require the following:

1. CPRF.EvalNonLin *and* CPRF.ConstrainEvalNonLin *are exactly the same algorithms. That is, they are expressed identically as circuits.*
2. *For any* $\mathbf{x}, \mathbf{y}$ *such that* $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ *and* $\mathsf{K} \xleftarrow{\$} \mathsf{CPRF.Setup}(\mathsf{pp})$ *where* $\mathsf{pp} \xleftarrow{\$} \mathsf{CPRF.Setup}(1^{\kappa})$*, we have*

$$\mathsf{CPRF.EvalLin}(\mathsf{K}, \mathbf{x}) = \mathsf{CPRF.ConstrainEvalLin}(\mathsf{CPRF.Constrain}(\mathsf{K}, \mathbf{y}), \mathbf{x}).$$

   Or equivalently, we have $\mathsf{K}^{\mathsf{int}}_{\mathbf{x}} = \mathsf{K}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}}$.
3. K, $\mathsf{K}^{\mathsf{con}}_{\mathbf{y}}$, $\mathsf{K}^{\mathsf{int}}_{\mathbf{x}}$, *and* $\mathsf{K}^{\mathsf{con\text{-}int}}_{\mathbf{y},\mathbf{x}}$ *are integer vectors. Also, for any* $\mathbf{x}, \mathbf{y} \in \mathcal{D}$*, algorithms* $\mathsf{CPRF.Constrain}(\cdot, \mathbf{y})$, $\mathsf{CPRF.EvalLin}(\cdot, \mathbf{x})$ *and* $\mathsf{CPRF.ConstrainEvalLin}(\cdot, \mathbf{x})$ *are linear functions over* $\mathbb{Z}$*. Moreover, their norms are at most* $\mathsf{poly}(\kappa, \ell, B)$*. (See Definition 3.1 for the definition of a norm of a linear function.)*

4. We have $\|\mathsf{K}^{\mathsf{int}}_{\mathbf{x}}\|_\infty = \mathsf{poly}(\kappa, \ell, B)$ *where* $\mathsf{pp} \xleftarrow{\$} \mathsf{CPRF.Setup}(1^\kappa)$, $\mathsf{K} \xleftarrow{\$} \mathsf{CPRF.Setup}(\mathsf{pp})$, *and* $\mathsf{K}^{\mathsf{int}}_{\mathbf{x}} := \mathsf{CPRF.EvalLin}(\mathsf{K}, \mathbf{x})$.

We stress that, in the second item above, we *do not* require that $\mathsf{CPRF.EvalLin}(\mathsf{K}, \mathbf{x})$ and $\mathsf{CPRF.ConstrainEvalLin}(\mathsf{CPRF.Constrain}(\mathsf{K}, \mathbf{y}), \mathbf{x})$ to be identical as (arithmetic) circuits; they are only required to have the same input/output. This is a crucial difference from the notion of conforming CPRF by Tsabary [46].

The key simulation property is essentially the same as defined by Tsabary [46].

Definition 4.3 (Key simulation). *The key simulation security is defined by the following game between an adversary* $\mathcal{A}$ *and a challenger:*

Setup: *At the beginning of the game, the challenger generates the public parameter* $\mathsf{pp} \xleftarrow{\$} \mathsf{CPRF.Setup}(1^\kappa)$ *and master key* $\mathsf{K} \xleftarrow{\$} \mathsf{CPRF.Gen}(\mathsf{pp})$, *and sends* pp *to* $\mathcal{A}$.

Queries: $\mathcal{A}$ *can adaptively make unbounded number of evaluation queries. Upon a query* $\mathbf{x} \in \mathcal{D}$, *the challenger returns* $r \xleftarrow{\$} \mathsf{CPRF.Eval}(\mathsf{pp}, \mathsf{K}, \mathbf{x})$.

Challenge Phase: *At some point,* $\mathcal{A}$ *makes a challenge query* $\mathbf{y}^* \in \mathcal{D}$. *Then the challenger uniformly picks* $\mathsf{coin} \xleftarrow{\$} \{0,1\}$. *If* $\mathsf{coin} = 0$, *then the challenger samples* $\widetilde{\mathsf{K}} \xleftarrow{\$} \mathsf{CPRF.Gen}(\mathsf{pp})$ *and returns* $\widetilde{\mathsf{K}}^{\mathsf{con}}_{\mathbf{y}^*} \xleftarrow{\$} \mathsf{CPRF.Constrain}(\widetilde{\mathsf{K}}, \mathbf{y}^*)$ *and otherwise returns* $\mathsf{K}^{\mathsf{con}}_{\mathbf{y}^*} \xleftarrow{\$} \mathsf{CPRF.Constrain}(\mathsf{K}, \mathbf{y}^*)$.

Queries: *After the challenge phase,* $\mathcal{A}$ *may continue to adaptively make unbounded number of evaluation queries. Upon a query* $\mathbf{x} \in \mathcal{D}$, *the challenger returns* $r \xleftarrow{\$} \mathsf{CPRF.Eval}(\mathsf{pp}, \mathsf{K}, \mathbf{x})$.

Guess: *Eventually,* $\mathcal{A}$ *outputs* $\widehat{\mathsf{coin}}$ *as a guess for* coin.

We say the adversary $\mathcal{A}$ *wins the game if* $\widehat{\mathsf{coin}} = \mathsf{coin}$ *and for any evaluation query* $\mathbf{x}$, *we have* $\langle \mathbf{x}, \mathbf{y}^* \rangle \neq 0$. *We require that for all PPT adversary* $\mathcal{A}$, $|\Pr[\mathcal{A}\ \textit{wins}] - 1/2| = \mathsf{negl}(\kappa)$ *holds.*

We note that the key simulation property easily follows from the adaptive single-key security of a standard CPRF.

Lemma 4.1 (Implicit in [46]). *If* Π_{CPRF} *is adaptively single-key secure, then it also satisfies the key simulation property.*

The uniformity requires that for any fixed input, the PRF value is uniform over the random choice of a key.

Definition 4.4 (Uniformity). *For all* $x \in \mathcal{D}$ *and* $r \in \mathcal{R}$, *we have*

$$\Pr[\mathsf{CPRF.Eval}(\mathsf{K}, \mathbf{x}) = r : \mathsf{pp} \xleftarrow{\$} \mathsf{CPRF.Setup}(1^\kappa), \mathsf{K} \xleftarrow{\$} \mathsf{CPRF.Setup}(\mathsf{pp})] = 1/|\mathcal{R}|.$$

We note that this is a very mild property, and we can generically add this property by applying a one-time pad. Namely, suppose that we include a uniform string $R \in \mathcal{R}$ in K, and slightly modify the evaluation algorithm so that it outputs

the XOR of the original output and R. Then it is clear that the resulting scheme satisfies the uniformity property. Moreover, it is easy to see that this conversion preserves the partial linear evaluation property and key simulation property. Combining this observation with Lemma 4.1, we obtain the following lemma.

Lemma 4.2. *If there exists a CPRF for inner-products that satisfies the partial linear evaluation property and the adaptive single-key security, then there exists an IPE-conforming CPRF that satisfies the partial linear evaluation, key simulation, and uniformity properties.*

Following [46], we use the following notations in Sec. 5:

- $U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{x}}$: A linear function computing $\mathsf{CPRF.EvalLin}(\cdot, \mathbf{x})$.
- $U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{y}}$: A linear function computing $\mathsf{CPRF.Constrain}(\cdot, \mathbf{y})$.
- $U^{\mathsf{lin}}_{\mathbf{y}\to\mathbf{x}}$: A linear function computing $\mathsf{CPRF.ConstrainEvalLin}(\cdot, \mathbf{x})$.
- $U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}$: A (not necesarily linear) function that computes $\mathsf{CPRF.EvalNonLin}(\mathsf{pp}, \cdot, \mathbf{x})$ $(= \mathsf{CPRF.ConstrainEvalNonLin}(\mathsf{pp}, \cdot, \mathbf{x}))$.

Note that $U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{x}}$ and $U^{\mathsf{lin}}_{\mathbf{y}\to\mathbf{x}} \circ U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{y}}$ are functionally equivalent for any $\mathbf{x}, \mathbf{y} \in \mathcal{D}$ such that $\langle \mathbf{x}, \mathbf{y} \rangle = 0$ by Item 2 of Definition 4.2.

4.2 Construction

We show that a variant of the LWE-based CPRF recently proposed by Davidson et al. [23] satisfies the required property. The scheme and security proof are largely the same as theirs. The detais can be found in the full version. Then we obtain the following theorem.

Theorem 4.1. *There exists an IPE-conforming CPRF assuming the LWE assumption with sub-exponential modulus size.*

5 Adaptively Secure IPE

In this section, we give a construction of an adaptively secure IPE scheme. The scheme will deal with inner products over vectors $\mathcal{D} := [-B, B]^\ell \subset \mathbb{Z}^\ell$ for any arbitrarily chosen $B(\kappa) = \mathsf{poly}(\kappa)$ and $\ell(\kappa) = \mathsf{poly}(\kappa)$. The main ingredient of the construction is a CPRF scheme $\Pi_{\mathsf{CPRF}} = (\mathsf{CPRF.Setup}, \mathsf{CPRF.Gen}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.ConstrainEval})$ for inner products over vectors in $\mathcal{D} := [-B, B]^\ell \subset \mathbb{Z}^\ell$ with IPE conforming property (See Definition 4.1). We assume that the size of the range $\mathcal{R}$ of the CPRF is super-polynomial in κ. We can instantiate such CPRF by the scheme in Theorem 4.1. To describe our scheme, we introduce the following parameters.

- For simplicity of notation, we assume that K, $\mathsf{K}^{\mathsf{int}}_{\mathbf{x}}$, and $\mathsf{K}^{\mathsf{con}}_{\mathbf{y}}$ are integer vectors with the same dimension $s(\kappa)$. This can be realized by choosing $s(\kappa)$ to be the maximum length of these vectors and padding the vectors with smaller dimensions by zeros. It is easy to see that the partial linear evaluation property and the security of the CPRF are preserved with this modification. Furthermore, by the efficiency of the CPRF, we can set $s(\kappa) = \mathsf{poly}(B(\kappa), \ell(\kappa)) = \mathsf{poly}(\kappa)$.

- We let $M(\kappa)$ be an upper bound on $\|\mathsf{K}^{\mathsf{int}}_{\mathbf{x}}\|_\infty$ and the norms of $U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{x}}$, $U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{y}}$, and $U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}$, where we refer to Definition 3.1 for the definition of norm for linear functions. By Items 3 and 4 of Definition 4.2, these quantities are bounded by $\mathsf{poly}(\kappa, \ell(\kappa), B(\kappa)) \le \mathsf{poly}(\kappa)$. We therefore can set $M(\kappa) = \mathsf{poly}(\kappa)$.
- We let $\eta(\kappa)$ to be the length of the output of the CPRF represented as a binary string. Namely, we have $\mathcal{R} \subseteq \{0,1\}^\eta$, where $\mathcal{R}$ is the range of the CPRF. We also assume $1/|\mathcal{R}| = \mathsf{negl}(\kappa)$ without loss of generality. If the CPRF does not satisfy the property, we can satisfy this by running $\omega(\kappa)$ number of the CPRF in parallel. We can easily see that this preserves the partial linear evaluation (Definition 4.2), key simulation security (Definition 4.3), and uniformity (Definition 4.4) properties.
- We let $d(\kappa)$ be an upper bound on the depth of the circuits $U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}$ and Eq_r, where $\mathsf{Eq}_r : \{0,1\}^\eta \to \{0,1\}$ is the circuit that on input $\widetilde{r} \in \{0,1\}^\eta$ returns 1 if and only if $r = \widetilde{r}$ for $r \in \{0,1\}^\eta$. We have that $d(\kappa) = \mathsf{poly}(\kappa, \ell(\kappa), B(\kappa)) \le \mathsf{poly}(\kappa)$ by the efficiency of the CPRF.

Then our IPE scheme $\Pi_{\mathsf{IPE}} = (\mathsf{IPE.Setup}, \mathsf{IPE.Enc}, \mathsf{IPE.KeyGen}, \mathsf{IPE.Dec})$ is described as follows. The lattice dimension $n(\kappa)$ and $m(\kappa)$, LWE modulus $q(\kappa)$, LWE noise distribution χ, Gaussian parameters $\tau_0(\kappa)$ and $\tau(\kappa)$, and width of noise $\Gamma(\kappa)$ in the scheme will be specified right after the description of the scheme.

$\mathsf{IPE.Setup}(1^\kappa)$**:** On input the security parameter 1^κ, it generates $\mathsf{pp}_{\mathsf{CPRF}} \xleftarrow{\$} \mathsf{CPRF.Setup}(1^\kappa)$, $\mathbf{k} \xleftarrow{\$} \mathsf{CPRF.Gen}(\mathsf{pp}_{\mathsf{CPRF}})$,[9] samples $(\mathbf{B}, \mathbf{B}^{-1}_{\tau_0}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m(s+1)}$, and $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$, and outputs $\mathsf{pp} := (\mathbf{B}, \mathbf{A}, \mathbf{v}, \mathsf{pp}_{\mathsf{CPRF}})$ and $\mathsf{msk} := (\mathbf{B}^{-1}_{\tau_0}, \mathbf{k})$.

$\mathsf{IPE.Enc}(\mathsf{pp}, \mathbf{y}, \mathsf{M})$**:** On input the public parameter pp, a vector $\mathbf{y} \in [-B, B]^\ell$, and a message $\mathsf{M} \in \{0,1\}$, it generates $\widetilde{\mathbf{k}} \xleftarrow{\$} \mathsf{CPRF.Gen}(\mathsf{pp}_{\mathsf{CPRF}})$ and samples $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}_0 \xleftarrow{\$} \chi^m$, $\mathbf{e}_1 \xleftarrow{\$} [-\Gamma, \Gamma]^{m(s+1)}$, and $e_2 \xleftarrow{\$} \chi$. It then computes $\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}} \leftarrow \mathsf{CPRF.Constrain}(\widetilde{\mathbf{k}}, C_{\mathbf{y}})$, sets

$$\mathbf{c}_0 = \mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top, \quad \mathbf{c}_1 = \mathbf{s}^\top [\mathbf{A}^{\mathsf{con}}_{\mathbf{y}} - (1(\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}})^\top) \otimes \mathbf{G}] + \mathbf{e}_1^\top, \quad c_2 = \mathbf{s}^\top \mathbf{v} + e_2 + \mathsf{M}\lfloor q/2 \rfloor$$

where $\mathbf{A}^{\mathsf{con}}_{\mathbf{y}} = \mathbf{A}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}}$ for $\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}} \leftarrow \mathsf{EvalLin}(U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{y}})$, and outputs $\mathsf{ct} := (\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}}, \mathbf{c}_0, \mathbf{c}_1, c_2)$.

$\mathsf{IPE.KeyGen}(\mathsf{pp}, \mathsf{msk}, \mathbf{x})$**:** On input the master secret key $\mathsf{msk} = (\mathbf{B}^{-1}_{\tau_0}, \mathbf{k})$ and a vector $\mathbf{x} \in [-B, B]^\ell$, it computes $r := \mathsf{CPRF.Eval}(\mathbf{k}, \mathbf{x})$,

$$\begin{aligned}
&\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}} \leftarrow \mathsf{EvalLin}(U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{x}}), && \mathbf{A}^{\mathsf{int}}_{\mathbf{x}} := \mathbf{A}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}},\\
&\mathbf{H}_{\mathbf{x}} \leftarrow \mathsf{EvalF}^{\mathsf{bd}}(1^M, U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}, \mathbf{A}^{\mathsf{int}}_{\mathbf{x}}), && \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}} := \mathbf{A}^{\mathsf{int}}_{\mathbf{x}}\mathbf{H}_{\mathbf{x}}\\
&\mathbf{H}_r \leftarrow \mathsf{EvalF}(\mathsf{Eq}_r, \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}}), && \mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r} := \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}}\mathbf{H}_r,
\end{aligned}$$

[9] We use $\mathbf{k}$ instead of K to denote the master secret key of CPRF for making it clear that it is a vector.

It then parses

$$\mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r} \to [\mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r,0} || \mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r,1}] \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{n \times m},$$

samples $\mathbf{u} \xleftarrow{\$} [\mathbf{B}||\mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r,1}]_\tau^{-1}(\mathbf{v})$ by using the trapdoor $\mathbf{B}_{\tau_0}^{-1}$, and outputs $\mathsf{sk}_{\mathbf{x}} := (r, \mathbf{u})$.

$\mathsf{IPE.Dec}(\mathsf{pp}, \mathsf{sk}_{\mathbf{x}}, \mathsf{ct}, \mathbf{y}, \mathbf{x})$**:** On input a secret key $\mathsf{sk}_{\mathbf{x}} = (r, \mathbf{u})$, a ciphertext $\mathsf{ct} = (\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}}, \mathbf{c}_0, \mathbf{c}_1, c_2)$, and vectors $\mathbf{y} \in [-B, B]^\ell$ and $\mathbf{x} \in [-B, B]^\ell$, it computes $\widetilde{\mathbf{k}}^{\mathsf{int}}_{\mathbf{x}} := U^{\mathsf{lin}}_{\mathbf{y}\to\mathbf{x}}(\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}})$ and $\widetilde{r} := U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}(\widetilde{\mathbf{k}}^{\mathsf{int}}_{\mathbf{x}})$ and aborts if $r = \widetilde{r}$. Otherwise, it computes

$$\overline{\mathbf{M}}_{\mathbf{y}\to\mathbf{x}} \leftarrow \mathsf{EvalLin}(U^{\mathsf{lin}}_{\mathbf{y}\to\mathbf{x}}), \quad \widehat{\mathbf{H}}_{\mathbf{x}} \leftarrow \mathsf{EvalFX}^{\mathsf{bd}}(1^M, U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}, \widetilde{\mathbf{k}}^{\mathsf{int}}_{\mathbf{x}}, \mathbf{A}^{\mathsf{int}}_{\mathbf{x}}),$$
$$\widehat{\mathbf{H}}_r \leftarrow \mathsf{EvalFX}(\mathsf{Eq}_r, \widetilde{r}, \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}})$$

where $\mathbf{A}^{\mathsf{int}}_{\mathbf{x}}$, and $\mathbf{A}^{\mathsf{eval}}_{\mathbf{x}}$ are computed as in $\mathsf{IPE.KeyGen}$. Then it computes $u := c_2 - [\mathbf{c}_0||\mathbf{c}_1\overline{\mathbf{M}}_{\mathbf{y}\to\mathbf{x}}\widehat{\mathbf{H}}_{\mathbf{x}}\widehat{\mathbf{H}}_r[\mathbf{0}_m||\mathbf{I}_m]^\top]\mathbf{u}$, and output 1 if $|u| \geq q/4$ and 0 otherwise.

A concrete parameter candidate and the correctness of the scheme are provided in the full version. We note that the parameters are set in a way that the LWE assumption with sub-exponential modulus size is believed to be hard. The security of our scheme is provided by the following theorem.

Theorem 5.1. *Under the hardness of the* $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\chi}}$ *problem,* Π_{IPE} *is adaptively payload-hiding if* Π_{CPRF} *is IPE-conforming.*

Proof. (sketc.h) We consider the following sequance of games between a valid adversary $\mathcal{A}$ and a challenger. In the following, we only give brief explanations on why each game is indistinguishable from the previous game. A full proof can be found in the full version. Below, let E_i denote the probability that $\widehat{\mathsf{coin}} = \mathsf{coin}$ holds in Game_i.

Game_0**:** This is the original adaptive security game. Specifically the game proceeds as follows:

- The challenger generates $\mathsf{pp}_{\mathsf{CPRF}} \xleftarrow{\$} \mathsf{CPRF.Setup}(1^\kappa)$, $\mathbf{k} \xleftarrow{\$} \mathsf{CPRF.Gen}(\mathsf{pp}_{\mathsf{CPRF}})$, samples $(\mathbf{B}, \mathbf{B}_{\tau_0}^{-1}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m(s+1)}$, and $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$, sets $\mathsf{pp} := (\mathbf{B}, \mathbf{A}, \mathbf{v}, \mathsf{pp}_{\mathsf{CPRF}})$, and gives pp to $\mathcal{A}$.
- Given pp, $\mathcal{A}$ makes unbounded number of key generation queries and one challenge query in arbitrary order.

Key Generation: When $\mathcal{A}$ makes a key generation query $\mathbf{x} \in [-B, B]^\ell$, the challenger computes $r := \mathsf{CPRF.Eval}(\mathbf{k}, \mathbf{x})$,

$$\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}} \leftarrow \mathsf{EvalLin}(U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{x}}), \qquad \mathbf{A}^{\mathsf{int}}_{\mathbf{x}} := \mathbf{A}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}},$$
$$\mathbf{H}_{\mathbf{x}} \leftarrow \mathsf{EvalF}^{\mathsf{bd}}(1^M, U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}, \mathbf{A}^{\mathsf{int}}_{\mathbf{x}}), \qquad \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}} := \mathbf{A}^{\mathsf{int}}_{\mathbf{x}}\mathbf{H}_{\mathbf{x}}$$
$$\mathbf{H}_r \leftarrow \mathsf{EvalF}(\mathsf{Eq}_r, \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}}), \qquad \mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r} := \mathbf{A}^{\mathsf{eval}}_{\mathbf{x}}\mathbf{H}_r,$$
$$\mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r,1} := \mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r}[\mathbf{0}_m||\mathbf{I}_m]^\top,$$

samples $\mathbf{u} \xleftarrow{\$} [\mathbf{B}||\mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r,1}]^{-1}_{\tau}(\mathbf{v})$ by using the trapdoor $\mathbf{B}^{-1}_{\tau_0}$, and returns $\mathsf{sk}_{\mathbf{x}} := (r, \mathbf{u})$ to $\mathcal{A}$.

Challenge: When $\mathcal{A}$ makes a challenge query $\mathbf{y}^*$, the challenger randomly picks $\mathsf{coin} \xleftarrow{\$} \{0,1\}$, generates $\widetilde{\mathbf{k}} \xleftarrow{\$} \mathsf{CPRF.Gen}(\mathsf{pp}_{\mathsf{CPRF}})$ and $\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}^*} \leftarrow \mathsf{CPRF.Constrain}(\widetilde{\mathbf{k}}, C_{\mathbf{y}^*})$, samples $\mathbf{s} \xleftarrow{\$} \mathbb{Z}^n_q$, $\mathbf{e}_0 \xleftarrow{\$} \chi^m$, $\mathbf{e}_1 \xleftarrow{\$} [-\Gamma, \Gamma]^{m(s+1)}$, $e_2 \xleftarrow{\$} \chi$, and sets

$$\mathbf{c}_0 = \mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top, \quad \mathbf{c}_1 = \mathbf{s}^\top [\mathbf{A}^{\mathsf{con}}_{\mathbf{y}^*} - (1(\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}^*})^\top) \otimes \mathbf{G}] + \mathbf{e}_1^\top,$$
$$c_2 = \mathbf{s}^\top \mathbf{v} + e_2 + \mathsf{coin}\lfloor q/2 \rfloor$$

where $\mathbf{A}^{\mathsf{con}}_{\mathbf{y}^*} = \mathbf{A}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}^*}$ for $\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}^*} \leftarrow \mathsf{EvalLin}(U^{\mathsf{lin}}_{\mathbf{k}\to\mathbf{y}^*})$, and returns $\mathsf{ct}^* := (\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}^*}, \mathbf{c}_0, \mathbf{c}_1, c_2)$ to $\mathcal{A}$.

– Finally, $\mathcal{A}$ outputs its guess $\widehat{\mathsf{coin}}$.

By the definition of E_0, the advantage of $\mathcal{A}$ is $|\Pr[\mathsf{E}_0] - 1/2|$.

Game_1**:** This game is identical to the previous game except that $\widetilde{\mathbf{k}}^{\mathsf{con}}_{\mathbf{y}^*}$ used in the challenge ciphertext is replaced with $\mathbf{k}^{\mathsf{con}}_{\mathbf{y}^*} \xleftarrow{\$} \mathsf{CPRF.Constrain}(\mathbf{k}, \mathbf{y}^*)$.

By a straightforward reduction to key-simulatability of the CPRF, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_0]| = \mathsf{negl}(\kappa)$.

Game_2**:** This game is identical to the previous game except that $\mathbf{A}$ is generated as $\mathbf{A} := \mathbf{B}\mathbf{R} + (1, \mathbf{k}^\top) \otimes \mathbf{G}$ where $\mathbf{R} \xleftarrow{\$} \{-1, 0, 1\}^{m \times m(s+1)}$.

By the leftover hash lemma, we have $|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_1]| = \mathsf{negl}(\kappa)$.

Game_3**:** This game is identical to the previous game except that $\mathbf{c}_1$ is generated as $\mathbf{c}_1 := \mathbf{c}_0\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}^*} + \mathbf{e}_1^\top$.

By Lemma 3.2, we can show that we have

$$\mathbf{c}_0\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}^*} + \mathbf{e}_1^\top = \mathbf{s}^\top [\mathbf{A}^{\mathsf{con}}_{\mathbf{y}^*} - (1(\mathbf{k}^{\mathsf{con}}_{\mathbf{y}^*})^\top) \otimes \mathbf{G}] + \mathbf{e}_0^\top \mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}^*} + \mathbf{e}_1^\top.$$

Moreover, by our choice of parameters, we can show that the distribution of $\mathbf{e}_0^\top \mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{y}^*} + \mathbf{e}_1^\top$ is statistically close to that of $\mathbf{e}_1^\top$. Therefore, we have $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(\kappa)$.

Game_4**:** This game is identical to the previous game except that in each key generation, $\mathbf{u}$ is generated as $\mathbf{u} \xleftarrow{\$} [\mathbf{B}||\mathbf{B}\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}}\widehat{\mathbf{H}}_{\mathbf{x}}\widehat{\mathbf{H}}_r[\mathbf{0}_m||\mathbf{I}_m]^\top + \mathbf{G}]^{-1}_{\tau}(\mathbf{v})$ where $\widehat{\mathbf{H}}_{\mathbf{x}} \leftarrow \mathsf{EvalFX}^{\mathsf{bd}}(1^M, U^{\mathsf{non\text{-}lin}}_{\mathbf{x}}, \mathbf{k}^{\mathsf{int}}_{\mathbf{x}}, \mathbf{A}^{\mathsf{int}}_{\mathbf{x}})$ and $\widehat{\mathbf{H}}_r \xleftarrow{\$} \mathsf{EvalFX}(\mathsf{Eq}_r, r, \mathbf{A}^{\mathsf{eq}}_{\mathbf{x}})$. We note that this can be done using $\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}}\widehat{\mathbf{H}}_{\mathbf{x}}\widehat{\mathbf{H}}_r[\mathbf{0}_m||\mathbf{I}_m]^\top$ instead of using $\mathbf{B}^{-1}_{\tau_0}$ if the norm of $\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}}\widehat{\mathbf{H}}_{\mathbf{x}}\widehat{\mathbf{H}}_r[\mathbf{0}_m||\mathbf{I}_m]^\top$ is small enough by a standard lattice trapdoor technique [1,37].

By Lemma 3.2 and Lemma 3.4, we can show that we have

$$\mathbf{B}\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}}\widehat{\mathbf{H}}_{\mathbf{x}}\widehat{\mathbf{H}}_r \begin{bmatrix}\mathbf{0}_m \\ \mathbf{I}_m\end{bmatrix} + \mathbf{G} = \mathbf{A}^{\mathsf{eq}}_{\mathbf{x},r,1}.$$

Moreover, by our choice of parameters, we can show that the norm of $\mathbf{R}\overline{\mathbf{M}}_{\mathbf{k}\to\mathbf{x}}\widehat{\mathbf{H}}_{\mathbf{x}}\widehat{\mathbf{H}}_r[\mathbf{0}_m||\mathbf{I}_m]^\top$ is small enough for sampling $\mathbf{u}$ in the above way. Therefore, $|\Pr[\mathsf{E}_4] - \Pr[\mathsf{E}_3]| = \mathsf{negl}(\kappa)$.

Game_5**:** This game is identical to the previous game except that $\mathbf{B}$ is generated as $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ instead of being generated with the trapdoor $\mathbf{B}_{\tau_0}^{-1}$. We note that this can be done since $\mathbf{B}_{\tau_0}^{-1}$ is no longer used due to the modification made in Game_4.

Since a matrix sampled with a trapdoor is almost uniformly distributed [25], we have $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(\kappa)$.

Game_6**:** This game is identical to the previous game except that $\mathbf{c}_0$ and c_2 are generated as $\mathbf{c}_0 \xleftarrow{\$} \mathbb{Z}_q^m$ and $c_2 \xleftarrow{\$} \mathbb{Z}_q$.
We can show that we have $|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_5]| = \mathsf{negl}(\kappa)$ by a straightforward reduction to the LWE assumption. Moreover, we have $\Pr[\mathsf{E}_6] = 1/2$ since no information of coin is given to $\mathcal{A}$ in this game.

Combining the above, we obtain $|\Pr[\mathsf{E}_0] - 1/2| = \mathsf{negl}(\kappa)$, which concludes the proof of Theorem 5.1.

6 Extensions to Other Adaptively Secure Predicate Encryptions

In this section, we show how to extend our IPE over the integers $\mathbb{Z}$ from the previous section to other types of ABEs. Specifically, we provide the following type of adaptive ABEs: IPE over $\mathbb{Z}_p$ for $p = \mathsf{poly}(\kappa)$ and fuzzy IBE for small and large universe. We achieve these extensions by encoding the attributes for one predicate to attributes in another predicate. Thus, our transformations are simple and the security reductions are straightforward. Since the former two generic constructions are almost folklore, we provide the formal description in the full version.

In the following, we first show how to encode a fuzzy predicate for large universe D (i.e., D is exponentially large) into a fuzzy predicate for small universe D' (i.e., D is polynomially large). First, we define some parameters and functions. Let $D = \{0,1\}^d$ be the alphabet domain of a FIBE for large universe where $d = \mathsf{poly}(\kappa)$. That is, $\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_L) \in D^L$ is a (row) vector of identities in FIBE for large universe. Let T be the threshold that satisfies $1 \leq T \leq L$. For a set S, a positive integer k, and a vector $\mathbf{x}, \mathbf{y} \in S^k$, let $\mathsf{HD}_{S^k}(\mathbf{x}, \mathbf{y})$ be the number of $i \in [k]$ such that $\mathbf{x}[i] \neq \mathbf{y}[i]$.

We use an error correcting code (ECC) $\mathsf{ECC} : D \to G^n$ such that $|G| = \mathsf{poly}(\kappa)$ and $n > d$. For simplicity, we use Reed-Solomon code [42]. More concretely, we consider $a \in D$ as a polynomial $p_a(X) := \sum_{i=1}^{d} a[i]X^{i-1}$ over $G := \mathbb{F}_q$ where q is a prime such that $n < q = \mathsf{poly}(\kappa)$ and a codeword is $f(a) = (p_a(1), ..., p_a(n))$. Then, $\mathsf{HD}_{G^n}(f(a), f(b)) \geq n-d+1$ holds for $a \neq b \in D$. We naturally extend the domain of f to D^L. That is, $\mathsf{ECC} : D^L \to (G^n)^L$

By the property of ECC, it holds that

$$0 \leq \mathsf{HD}_{D^L}(\mathbf{a}, \mathbf{b}) \leq L - T \Longrightarrow 0 \leq \mathsf{HD}_{G^{nL}}(f(\mathbf{a}), f(\mathbf{b})) \leq (L-T)n$$
$$L - T + 1 \leq \mathsf{HD}_{D^L}(\mathbf{a}, \mathbf{b}) \leq L \Longrightarrow (L-T+1)(n-d+1) \leq \mathsf{HD}_{G^{nL}}(f(\mathbf{a}), f(\mathbf{b})) \leq Ln$$

for two identities $\mathbf{a} = (a_1, ..., a_L), \mathbf{b} = (b_1, ..., b_L) \in D^L$. Therefore, for a fixed T, we will set (n, d) as

$$(L - T)n < (L - T + 1)(n - d + 1). \tag{15}$$

This allows us to argue that a "gap" exists in the hamming distance defined over polynomially large domains if there is a "gap" in the hamming distance defined over exponentially large domains.

Notably, we can reduce a fuzzy predicate $\mathsf{P}_{\mathsf{exp}}$ for exponentially large alphabet strings to a fuzzy predicate $\mathsf{P}_{\mathsf{poly}}$ for polynomially large alphabet strings. That is, we first encode $\mathbf{a} \in D^L$ into $f(\mathbf{a}) \in G^{nL}$ by using an ECC. Then, if the threshold of $\mathsf{P}_{\mathsf{exp}}$ is T, we set the threshold of $\mathsf{P}_{\mathsf{poly}}$ to be Tn. Lastly, we set $n > (d-1)(L-T+1)$ to satisfy Equation (15). Notice n is some polynomial in κ since $d = \mathsf{poly}(\kappa)$, $L = \mathsf{poly}(\kappa)$, and $0 \leq T \leq L$.

Translating the above encoding technique to the ABE context is straightforward and is omitted to the full version.

Acknowledgement. We thank anonymous reviewers for their helpful comments. The first and the third authors were supported by JST CREST Grant Number JPMJCR19F6 and JSPS KAKENHI Grant Number JP19H01109.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or fuzzy ibe) from lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 280–297. Springer, Heidelberg (2012)
3. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional encryption for inner product predicates from learning with errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_2
4. Agrawal, S., Libert, B., Stehlé, D.: Fully secure functional encryption for inner products, from standard assumptions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. Part III, volume 9816 of LNCS, pp. 333–362. Springer, Heidelberg (2016)
5. Attrapadung, N.: Dual system encryption via doubly selective security: framework, fully secure functional encryption for regular languages, and more. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 557–577. Springer, Heidelberg (2014)
6. Attrapadung, N.: Dual system encryption framework in prime-order groups via computational pair encodings. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 591–623. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_20
7. Attrapadung, N., Matsuda, T., Nishimaki, R., Yamada, S., Yamakawa, T.: Constrained PRFs for NC^1 in traditional groups. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. Part II, volume 10992 of LNCS, pp. 543–574. Springer, Heidelberg (2018)

8. Boneh, D., Boyen, X.: Efficient selective identity-based encryption without random oracles. J. Cryptol. **24**(4), 659–693 (2011)
9. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. SIAM J. Comput. **32**(3), 586–615 (2003)
10. Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014)
11. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
12. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. Part II, volume 8270 of LNCS, pp. 280–300. Springer, Heidelberg (2013)
13. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014)
14. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC, pp. 575–584. ACM Press, June 2013
15. Brakerski, Z., Tsabary, R., Vaikuntanathan, V., Wee, H.: Private constrained PRFs (and more) from LWE. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. Part I, volume 10677 of LNCS, pp. 264–302. Springer, Heidelberg (2017)
16. Brakerski, Z., Vaikuntanathan, V.: Constrained key-homomorphic PRFs from standard lattice assumptions. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 1–30. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_1
17. Brakerski, Z., Vaikuntanathan, V.: Circuit-ABE from LWE: unbounded attributes and semi-adaptive security. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. Part III, volume 9816 of LNCS, pp. 363–384. Springer, Heidelberg (2016)
18. Canetti, R., Chen, Y.: Constraint-hiding constrained PRFs for NC^1 from LWE. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10210, pp. 446–476. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_16
19. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. J. Cryptol. **25**(4), 601–639 (2012)
20. Chen, J., Gay, R., Wee, H.: Improved dual system ABE in prime-order groups via predicate encodings. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. Part II, volume 9057 of LNCS, pp. 595–624. Springer, Heidelberg (2015)
21. Chen, Y., Vaikuntanathan, V., Wee, H.: GGH15 beyond permutation branching programs: proofs, attacks, and candidates. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. Part II, volume 10992 of LNCS, pp. 577–607. Springer, Heidelberg (2018)
22. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45325-3_32
23. Davidson, A., Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Adaptively secure constrained pseudorandom functions in the standard model. CRYPTO **2020**, 111 (2020)
24. Gentry, C.: Practical identity-based encryption without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_27

25. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC, pp. 197–206. ACM Press, May 2008
26. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. Part I, volume 8042 of LNCS, pp. 75–92. Springer, Heidelberg (2013)
27. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption with bounded collusions via multi-party computation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 162–179. Springer, Heidelberg (2012)
28. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. J. ACM **62**(6), 45:1–45:33 (2015)
29. Gorbunov, S., Vaikuntanathan, V., Wichs, D.: Leveled fully homomorphic signatures from standard lattices. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th ACM STOC, pp. 469–477. ACM Press, June 2015
30. Goyal, R., Koppula, V., Waters, B.: Semi-adaptive security and bundling functionalities made generic and easy. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9986, pp. 361–388. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_14
31. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: Umans, C. (ed.) 58th FOCS, pp. 612–621. IEEE Computer Society Press, October 2017
32. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.)ACM CCS 2006, pp. 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309
33. Katsumata, S., Yamada, S.: Non-zero inner product encryption schemes from various assumptions: LWE, DDH and DCR. In: Lin, D., Sako, K. (eds.) PKC 2019. Part II, volume 11443 of LNCS, pp. 158–188. Springer, Heidelberg (2019)
34. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. J. Cryptol. **26**(2), 191–224 (2013)
35. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A-R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 669–684. ACM Press, November 2013
36. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: attribute-based encryption and (Hierarchical) inner product encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_4
37. Micciancio, D., Peikert, C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)
38. Okamoto, T., Takashima, K.: Adaptively attribute-hiding (hierarchical) inner product encryption. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 591–608. Springer, Heidelberg (2012)
39. Okamoto, T., Takashima, K.: Fully secure functional encryption with a large class of relations from the decisional linear assumption. J. Cryptol. **32**(4), 1491–1573 (2019)
40. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher, M. (ed.) 41st ACM STOC, pp. 333–342. ACM Press, May / June 2009

41. Peikert, C., Shiehian, S.: Privately constraining and programming PRFs, the LWE way. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. Part II, volume 10770 of LNCS, pp. 675–701. Springer, Heidelberg (2018)
42. Reed, I.S., Solomon, G.: Polynomial codes over certain finite fields. SIAM J. Comput. **8**(2), 300–304 (1960)
43. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 34:1–34:40 (2009)
44. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
45. Shi, E., Bethencourt, J., Chan, H.T.-H., Song, X.D., Perrig, A.: Multi-dimensional range query over encrypted data. In: 2007 IEEE Symposium on Security and Privacy, pp. 350–364. IEEE Computer Society Press, May 2007
46. Tsabary, R.: Fully secure attribute-based encryption for t-CNF from LWE. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. Part I, volume 11692 of LNCS, pp. 62–85. Springer, Heidelberg (2019)
47. Wang, G., Wan, M., Liu, Z., Dawu, G.: Dual system in lattice: fully secure abe from lwe assumption. IACR Cryptol. ePrint Arch. **2020**, 64 (2020)
48. Waters, B.: Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)
49. Wee, H.: Dual system encryption via predicate encodings. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 616–637. Springer, Heidelberg (2014)
50. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: Umans, C. (ed.) 58th FOCS, pp. 600–611. IEEE Computer Society Press, October 2017