Asymptotically Compact Adaptively Secure Lattice IBEs and Verifiable Random Functions via Generalized Partitioning Techniques

Shota Yamada*[1]

[1]National Institute of Advanced Industrial Science and Technology (AIST).

June 5, 2017

Abstract

In this paper, we focus on the constructions of adaptively secure identity-based encryption (IBE) from lattices and verifiable random function (VRF) with large input spaces. Existing constructions of these primitives suffer from low efficiency, whereas their counterparts with weaker guarantees (IBEs with selective security and VRFs with small input spaces) are reasonably efficient. We try to fill these gaps by developing new partitioning techniques that can be performed with compact parameters and proposing new schemes based on the idea.

- We propose new lattice IBEs with poly-logarithmic master public key sizes, where we count the number of the basic matrices to measure the size. Our constructions are proven secure under the LWE assumption with polynomial approximation factors. They achieve the best asymptotic space efficiency among existing schemes that depend on the same assumption and achieve the same level of security.

- We also propose several new VRFs on bilinear groups. In our first scheme, the size of the proofs is poly-logarithmic in the security parameter, which is the smallest among all the existing schemes with similar properties. On the other hand, the verification keys are long. In our second scheme, the size of the verification keys is poly-logarithmic, which is the smallest among all the existing schemes. The size of the proofs is sub-linear, which is larger than our first scheme, but still smaller than all the previous schemes.

1 Introduction

1.1 Background

In cryptography, we define appropriate security notions for cryptographic primitives, in order to capture real world attacks. For a cryptographic scheme to be useful, it is desirable that the scheme achieves security notions as realistic as possible. However, since natural and realistic security notions are hard to achieve in general, we sometimes are only able to prove ad-hoc and unrealistic security notions. Even when proving the former is possible, it sometimes comes with the cost of longer parameters or stronger assumptions. In this paper, we focus on two such primitives: identity-based encryption (IBE) and verifiable random function (VRF).

*yamada-shota@aist.go.jp

Identity-Based Encryption. IBE [Sha85] is a generalization of public key encryption where the public key of a user can be any arbitrary string such as an e-mail address. The first realizations of IBE are given by [SOK00, BF01] on groups equipped with bilinear maps. Since then, realizations from bilinear maps [BB04a, BB04b, Wat05, Gen06, Wat09], from quadratic residues modulo composite [Coc01, BGH07], and from lattices [GPV08, CHKP10, ABB10a, Boy10] have been proposed.

Among the existing lattice IBE schemes in the standard model, the most efficient one is in [ABB10a]. However, the scheme only satisfies selective security, where an adversary must declare at the start of the game which identity it intends to target. Although schemes with a much more realistic adaptive security (or equivalently, full security) are known [CHKP10, ABB10a, Boy10], they are not as efficient as the aforementioned selectively secure scheme. In particular, all these schemes require master public keys longer by a factor $O(\lambda)$ than the selectively secure one, where λ is the security parameter. This stands in sharp contrast to pairing-based settings, in which we have adaptively secure IBE schemes [Wat09, CLL+12, JR13] that are as efficient as selectively secure ones [BB04a], up to a small constant factor.

There have been several studies that aim at reducing the sizes of the parameters in adaptively secure lattice IBEs [Yam16, AFL16, ZCZ16, KY16]. However, current state of affairs are not satisfactory. These schemes are either based on stronger assumptions [Yam16, KY16], or require still long public parameters [Yam16, KY16, AFL16], or only achieves weaker security guarantee [ZCZ16].

Verifiable Random Function. The notion of VRF was introduced by Micali, Rabin, and Vadhan [MRV99]. A VRF $\mathsf{V}_{\mathsf{sk}}(\cdot)$ is a pseudorandom function with the additional property that it is possible to create a non-interactive and publicly verifiable proof π that a given function value Y was computed correctly as $Y = \mathsf{V}_{\mathsf{sk}}(X)$. Since the introduction of this notion, several realizations have been proposed [MRV99, Lys02, Dod03, DY05, ACF09]. All these constructions only allow a polynomially bounded input space, or do not achieve full adaptive security without complexity leveraging, or are based on an interactive complexity assumption. Following [HJ16], in the sequel, we will say that a VRF has *all the desired properties*, if it has an exponential-sized input space and a proof of full adaptive security under a non-interactive complexity assumption.

The first VRF scheme with all the desired properties was proposed by Hohenberger and Waters [HW10]. Later, constructions from weaker assumptions have been studied [BMR10, ACF14, Jag15, HJ16]. Notably, the scheme in [HJ16] is secure under the standard decisional linear assumption. On the other hand, there has not been improvement on the efficiency since [HW10]. Namely, all existing VRF schemes with all the desired properties require $O(\lambda)$ group elements both in the verification keys and proofs. This is much more inefficient than the scheme with a polynomial-size input space [DY05], which only requires $O(1)$ group elements for both.

The Gaps in Efficiency. As we have seen, there is a distinct gap in efficiency between the state of the art schemes and the desired schemes. Namely, both in lattice IBEs and VRFs, we loose efficiency when we want to achieve stronger security notions. This loss in efficiency is an artifact of the security proofs. Most of the schemes use the partitioning technique based on (an analogue of) Waters' hash [Wat05] or admissible hash functions [BB04b] to achieve adaptive security. However, these techniques typically require long parameters. The powerful framework of dual system encryption methodology, which was introduced by Waters [Wat09], does not seem to be applicable for these settings. In particular, we do not have lattice analogue of the dual system approach yet. Furthermore, the uniqueness property required for VRF seems to contradict the algebraic structure required to apply the dual system approach, as pointed out in [Jag15, HJ16].

1.2 Our Contributions

In this paper, we try to fill the above gaps by generalizing the partitioning technique and proposing new schemes with improved (asymptotic) efficiency. To do so, we first introduce the notion of *partitioning functions*, which can be thought of as a generalization of the standard admissible hash functions [BB04b, CHKP10, FHPS13, Jag15]. The notion of partitioning functions abstracts out the information theoretic properties that are required to perform the partitioning technique in the security proofs for IBE and VRF. Then, we propose two new partitioning functions that can be constructed by much more compact parameters than prior admissible hash functions. Our first construction is obtained by compressing the expression of the existing admissible hash functions by introducing a novel encoding technique, whereas the second construction is based on affine functions over a random modulus. We call the first partitioning function $\mathsf{F}_{\mathsf{MAH}}$ and the second $\mathsf{F}_{\mathsf{AFF}}$, where MAH and AFF stand for modified admissible hash function and affine function respectively. These functions provide us a framework to perform the security proofs in a more space efficient manner than previous ones.

One thing to note is that in order to use them to construct IBE and VRF schemes, we need a certain level of homomorphic capability on the underlying algebraic structures. In the lattice setting, we can implement the idea by carefully applying the powerful fully key homomorphic techniques of [BGG$^+$14, GV15]. On the other hand, in the bilinear group setting, this technique may be inapplicable since we only have very limited amount of homomorphic capabilities. Namely, given group elements, which can be seen as encodings of the corresponding discrete logarithms, we can only compute encodings corresponding to quadratic multi-variate polynomials on them. However, in the special case of VRF, since the evaluator has full access to the secret key, it can evaluate any homomorphism on them to compute the function value. Based on this observation, we can implement the idea in this setting as well.

New Lattice IBE Schemes. Based on the new partitioning functions, we propose two new adaptively secure lattice IBE schemes. For the overview and comparison, we refer to Table 1. Both our schemes achieve the best asymptotic space efficiency among existing schemes with the same assumption and security notion. In particular, the number of basic matrices in the master public keys are only polylogarithmic. Furthermore, the sizes of the ciphertexts and private keys are optimal, in the sense that they match those of the selectively secure schemes [ABB10a, Boy10] up to a constant factor.

- In our first scheme, the master public key consists of $\omega(\log^2 \lambda)$ basic matrices[*], which is the smallest among all the previous schemes. The security of the scheme can be shown from the LWE assumption with approximation factor $\tilde{O}(n^{11})$, where n is the dimension of the lattices.

- In our second scheme, the master public key consists of only $\omega(\log \lambda)$ basic matrices, which is even smaller than the one above. The security of the scheme can be shown from the LWE assumption with approximation factors $\mathsf{poly}(n)$, where $\mathsf{poly}(n)$ is some fixed polynomial that is determined by the depth of the circuit computing a certain function.

We constructed the above schemes in a modular way. We first define the notion of *compatible algorithms* for partitioning functions. Then, we propose a generic construction of an IBE scheme

[*] In our paper, when we say that the size of a parameters is $\omega(f(\lambda))$, it means that the parameter can be set to be *any* (polynomially bounded) function that grows faster than $f(\lambda)$. The parameter can be as small as one wants, as long as it does not violate the lower-bound given by the ω-notation. In this case, we can choose the number of the matrices to be $\Theta(\log^3 \lambda)$ or even $\Theta(\log^2 \lambda \cdot \log\log\log \lambda)$ for instance.

Table 1: Comparison of Adaptively Secure Lattice IBE Schemes

Schemes	$\lvert\mathsf{mpk}\rvert$ # of $\mathbb{Z}_q^{n\times m}$ mat.	$\lvert\mathsf{ct}\rvert$, $\lvert\mathsf{sk}\rvert$ # of $\mathbb{Z}_q^{m}$ vec.	LWE param $1/\alpha$	Reduction Cost	Remarks
[CHKP10]	$O(\lambda)$	$O(\lambda)$	$\tilde{O}(n^{1.5})$	$O(\epsilon^{\nu+1}/Q^{\nu})^\ddagger$	
[ABB10a]+[Boy10]	$O(\lambda)$	$O(1)$	$\tilde{O}(n^{5.5})$	$O(\epsilon^2/qQ)$	
[Yam16]	$O(\lambda^{1/\mu})^\dagger$	$O(1)$	$n^{\omega(1)}$	$O(\epsilon^{\mu+1}/kQ^{\mu})^\dagger$	
[ZCZ16]	$O(\log Q)$	$O(1)$	$\tilde{O}(Q^2\cdot n^{6.5})$	$O(\epsilon/kQ^2)$	Q-bounded
[AFL16]*	$O(\lambda/\log^2\lambda)$	$O(1)$	$\tilde{O}(n^6)$	$O(\epsilon^2/qQ)$	
[BL16]	$O(\lambda)$	$O(1)$	$\mathsf{superpoly(n)}$	$O(\lambda)$	
[KY16]**	$O(\lambda^{1/\mu})^{**\dagger}$	$O(1)$	$O(n^{2.5+2\mu})^\dagger$	$O(\lambda^{\mu-1}\epsilon^{\mu}/Q^{\mu})^{\mu+1})^\dagger$	Ring-based
Sec. 5.4 + $\mathsf{F}_{\mathsf{MAH}}$ $(\eta=\log^2\lambda)$.	$O(\log^3\lambda)$	$O(1)$	$\tilde{O}(n^{11})$	$O(\epsilon^{\nu+1}/Q^{\nu})^\ddagger$	
Sec. 5.4 + $\mathsf{F}_{\mathsf{AFF}}$ $(\eta=\log^2\lambda)$.§	$O(\log^2\lambda)$	$O(1)$	$\mathsf{poly}(n)$	$O(\epsilon^2/k^2Q)$	Need [BCH86, Bar89]

We compare with adaptively secure IBE schemes under the LWE assumption in the standard model. $\lvert\mathsf{mpk}\rvert$, $\lvert\mathsf{ct}\rvert$, and $\lvert\mathsf{sk}_{\mathsf{ID}}\rvert$ show the size of the master public keys, ciphertexts, and private keys, respectively. To measure the space efficiency, we count the number of basic components. Q and ϵ denote the number of key extraction queries and the advantage, respectively. $\mathsf{poly}(n)$ (resp. $\mathsf{superpoly}(n)$) represents fixed but large polynomial (super-polynomial) that does not depend Q and ϵ. To measure the reduction cost, we show the advantage of the LWE algorithm constructed from the adversary against the corresponding IBE scheme. To be fair, we calculate the reduction cost by employing the technique of Bellare and Ristenpart [BR09] for all schemes.

† $\mu \in \mathbb{N}$ is a constant number that can be chosen arbitrary. Since the reduction cost degrades exponentially as μ grows, we would typically set μ very small (e.g., $\mu = 2$ or 3).

‡ $\nu > 1$ is the constant satisfying $c = 1 - 2^{-1/\nu}$, where c is the relative distance of the underlying error correcting code $C : \{0,1\}^k \to \{0,1\}^\ell$. We can take ν as close to 1 as one wants, by choosing $c < 1/2$ appropriately and make ℓ large enough (See Appendix E.1 of [Gol08]).

* They also propose a variant of the scheme with constant-size master public key assuming the exponentially secure collision resistant hash function. Since the use of the exponential assumption can be considered as a certain kind of the complexity leveraging, we do not include the variant in the table.

** The scheme can only be instantiated over the rings $R_q = \mathbb{Z}_q[X]/(X^n+1)$. To measure the size of mpk we count the number of the basic vectors, instead of the basic matrices.

§ The key generation and encryption algorithm of the scheme involves the heavy step of computing the description of the division circuit in NC^1 using the result of [BCH86] and converting it into a branching program by invoking the Barrington's theorem [Bar89].

from a partitioning function with its associating compatible algorithms. We obtain our first scheme by instantiating this framework with $\mathsf{F}_{\mathsf{MAH}}$ and its compatible algorithms. We obtain our second scheme by instantiating it with $\mathsf{F}_{\mathsf{AFF}}$.

New VRF Schemes. We also obtain the following three new VRF schemes with all the desired properties. For the overview and comparison, we refer to Table 2. All our schemes are constructed on bilinear groups and proven secure under the L-DDH assumption,[†] as is the same as most of the previous schemes [ACF14, BMR10, Jag15]. In the following, to measure the sizes of the proofs and verification keys, we count the number of group elements. Note that in all existing VRF schemes with all the desired properties [HW10, ACF14, BMR10, Jag15, HJ16], the sizes of the verification keys and proofs are $O(\lambda)$.

- Our first scheme is based on $\mathsf{F}_{\mathsf{MAH}}$, and is parametrized by several parameters, which control the tradeoffs of the efficiency. In certain parameter settings, the scheme achieves the smallest proof-size among all existing VRF schemes that satisfy all the desired properties. The size

[†] The L-DDH assumption says that given elements g, h, $g^\alpha, \ldots, g^{\alpha^L}$ in a bilinear group, $e(g,h)^{1/\alpha}$ is pseudorandom for any PPT adversary.

Table 2: Comparison of VRF Schemes with All The Desired Properties

Schemes	$\lvert\mathsf{vk}\rvert$ (# of $\mathbb{G}$)	$\lvert\mathsf{sk}\rvert$ (# of $\mathbb{Z}_p$)	$\lvert\pi\rvert$ (# of $\mathbb{G}$)	Assumption	Reduction Cost
[ACF14]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^{\nu})^{\dagger}$
[BMR10]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$-DDH	$O(\epsilon/\lambda)$
[HW10]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(Q\lambda/\epsilon)$-DDHE	$O(\epsilon^2/\lambda Q)$
[Jag15]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\log{(Q/\epsilon)})$-DDH	$O(\epsilon^{\nu+1}/Q^{\nu})^{\dagger}$
[HJ16]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	DLIN	$O(\epsilon^{\nu+1}/\lambda Q^{\nu})^{\dagger}$
Sec. 6.1 ($\ell_1 = \ell, \ell_2 = 1, \eta = \log^2 \lambda$).	$O(\lambda \log^2 \lambda)$	$O(\log^2 \lambda)$	$O(\log^2 \lambda)$	$\tilde{O}(\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^{\nu})^{\dagger}$
Sec. 6.3 ($\ell_1 = \ell_2 = \sqrt{\ell}, \eta = \log^2 \lambda$).	$O(\log^2 \lambda)$	$O(\log^2 \lambda)$	$O(\sqrt{\lambda} \log^2 \lambda)$	$\tilde{O}(\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^{\nu})^{\dagger}$
Appendix C ($\eta = \log^2 \lambda$)	$O(\log^2 \lambda)$	$O(\log^2 \lambda)$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$-DDH	$O(\epsilon^2/\lambda^2 Q)$

We compare VRF schemes with all the desired properties. $|\mathsf{vk}|$, $|\mathsf{sk}|$, and $|\pi|$ show the size of the verification keys, secret keys, and proofs, respectively. To measure $|\mathsf{vk}|$ and $|\pi|$ (resp. $|\mathsf{sk}|$), we count the number of group elements (resp. elements in $\mathbb{Z}_p$). Q and ϵ denote the number of evaluation queries and the advantage, respectively. $\mathsf{poly}(\lambda)$ represents fixed polynomial that does not depend Q and ϵ. To measure the reduction cost, we show the advantage of the algorithm that solves the problem (which is L-DDH for some L except for [HJ16]) constructed from the adversary against the corresponding VRF scheme. To be fair, we measure the reduction cost by employing the technique of Bellare and Ristenpart [BR09] for all schemes.

† ν is the constant satisfying $c = 1 - 2^{-1/\nu}$, where c is the relative distance of the underlying error correcting code $C : \{0,1\}^k \to \{0,1\}^{\ell}$. We can take ν as close to 1 as one wants, by choosing $c < 1/2$ appropriately and make ℓ large enough (See Appendix E.1 of [Gol08]).

of the proofs is $\omega(\log \lambda)$, whereas the size of the verification keys is $\omega(\lambda \log \lambda)$. The security is proven from the L-DDH assumption with $L = \tilde{O}(\lambda)$.

- Our second scheme is obtained by setting the parameters appropriately in our first scheme and modifying it slightly. The scheme achieves the smallest verification-key-size among all existing schemes with all the desired properties. The size of the verification keys is $\omega(\log \lambda)$, whereas the size of the proofs is $\omega(\sqrt{\lambda} \log \lambda)$. The size of the proofs is larger than our first scheme, but still smaller than all the previous schemes. The security is proven from the L-DDH assumption with $L = \tilde{O}(\lambda)$.
- Our third scheme is based on $\mathsf{F}_{\mathsf{AFF}}$. The size of the verification keys and the proofs are $\omega(\log \lambda)$ and $\mathsf{poly}(\lambda)$, respectively. The security of the scheme is proven from the L-DDH assumption with $L = \mathsf{poly}(\lambda)$. Here, $\mathsf{poly}(\lambda)$ is some fixed polynomial that is determined by the depth of the circuit computing a certain function.

Note that the main advantage of the third scheme over our first and second schemes is that the security reduction is tighter.

Finally we note that even though our lattice IBE schemes achieve the best asymptotic space efficiency, it might not outperform [ABB10a, Boy10] in practical parameter settings, due to the large poly-logarithmic factors and the heavy encryption algorithm. The construction of truly efficient adaptively secure lattice IBE still remains open.

Comparison with the Dual System Encryption Methodology. The dual system encryption methodology [Wat09, LW10] is a very powerful tool to prove the adaptive security of IBE and even advanced cryptographic primitives such as attribute-based encryption [LOS$^+$10]. However, currently, the technique is not available in several settings. These include lattice-based cryptography and the construction of VRF. We notice that relatively high level of homomorphic capabilities are available in these settings and show that the partitioning technique can be performed more compactly by exploiting this fact. Our technique is somewhat limited in the sense that it requires some homomorphic capabilities and may not be available without them. However, in the settings where our technique does not apply, the dual system encryption methodology may apply. In this sense, they have mutual complementary relationship.

1.3 Related Works

Related Works on Lattice IBE. Yamada [Yam16] used the fully key homomorphic technique of [BGG+14] and asymptotically reduced the size of the master public key. However, it required super-polynomial size modulus. The subsequent work by Katsumata et al. [KY16] showed that for the ring version of Yamada's scheme, it is possible to prove the security for polynomial-size modulus. The scheme by Apon et al. [AFL16] also proposed a scheme with shorter master public keys using a different technique. These schemes require larger number of matrices in the master public keys than ours. The scheme by Zhang et al. [ZCZ16] achieved shorter master public key size than ours, however at the cost of a weaker security guarantee. In particular, their scheme only achieves Q-bounded security, i.e., that the security of the scheme is not guaranteed any more if the number of key extraction queries that the adversary makes exceeds Q, where Q is a parameter that must be *determined at the setup phase* of the scheme. This restriction cannot be removed by just making Q super-polynomial, since the encryption algorithm of the scheme runs in time proportional to Q. Finally, Boyen and Li [BL16] proposed the first lattice IBE schemes with tight security reductions, where the schemes require long master public keys.

Related Works on VRF. Very recently, several works [GHKW17, Bit17, BGJS17] showed generic constructions of VRF from simpler cryptographic primitives. These constructions lead to VRF schemes from various assumptions, including schemes without bilinear maps. However, they cannot be efficiently instantiated because they require general NIWI and constrained PRF (for admissible hash). On the other hand, we focus on the efficient constructions of VRF from the specific number theoretic assumption. While our results are orthogonal to theirs, our definition of partitioning function is very similar to that of the "partitioning scheme" in the independent and concurrent work by Bitansky [Bit17].

2 Technical Overview

2.1 A Twist on the Admissible Hash

We first start with the review of the adaptively secure IBE schemes that use the admissible hash function [BB04b, CHKP10]. The security proofs of these schemes are based on the partitioning technique, a proof methodology that allows to secretly partition the identity space into two sets of exponential size, the uncontrolled set and the controlled set, so that there is a noticeable probability that the adversary's key extraction queries fall in the controlled set and the challenge identity falls in the uncontrolled set. Whether the identity is controlled or uncontrolled is determined by a function $\mathsf{F}_{\mathsf{ADH}}$ that on input a secret randomness K chosen during the simulation and an identity ID outputs 0 or 1. Here, 0 (resp. 1) indicates that ID is in the uncontrolled set (resp. controlled set). Concretely, the partitioning is made by the following specific function:

$$\mathsf{F}_{\mathsf{ADH}}(K, \mathsf{ID}) = \begin{cases} 0, & \text{if} \quad \forall i \in [\ell] : \quad C(\mathsf{ID})_i = K_i \quad \vee \quad K_i = \perp \\ 1, & \text{otherwise} \end{cases}$$

Figure 1: Pictorial explanation of the definition of S and T.

$K =$	$\perp$	$\perp$	1	$\perp$	0	$\perp$	$\perp$
	1	3	⑤	7	9	11	13
	2	4	6	8	⑩	12	14
$\mathsf{T} =$	$\{$		5,		10,		$\}$

$C(X)=$	0	1	1	0	0	1	0
	1	③	⑤	7	9	⑪	13
	②	4	6	⑧	⑩	12	⑭
$\mathsf{S}(X) =$	$\{2,$	3,	5,	8,	10,	11,	$14\}$

where $C(\cdot)$ is a public function that maps an identity to a bit string in $\{0,1\}^\ell$ and K is a string in $\{0,1,\perp\}^\ell$. $C(\mathsf{ID})_i$ and K_i represent the i-th bit of $C(\mathsf{ID})$ and the i-th component of K, respectively. In [BB04b, CHKP10], the master public keys are sufficiently long so that we can embed the secret randomness K into them in a component-wise manner in the security proof. Since $\ell = \Theta(\lambda)$, where λ is the security parameter, this results in large master public keys containing $O(\lambda)$ basic components. Due to the similar reasons, all constructions of VRFs using admissible hash functions [ACF14, BMR10, Jag15, HJ16] also suffer from large public parameters.

Our first step to address the problem is to observe that K is very "sparse" in the sense that it conveys only a small amount of information compared to its length. In the simulation, K is chosen uniformly at random from $\{0,1,\perp\}^\ell$, with $O(\log(Q/\epsilon))$ components being not $\perp$, where Q and ϵ are the number of key extraction queries and the advantage of the adversary, respectively. Since we assume an adversary that makes polynomial number of key extraction queries and has non-negligible advantage in the security proof, we have $O(\log(Q/\epsilon)) = O(\log\lambda)$. This means that $K_i = \perp$ for most $i \in [\ell]$.

Our key idea is to encode K into a much shorter bit-string. For $K \in \{0,1,\perp\}^\ell$, let us consider a set $\mathsf{T} \subseteq \{1,2,\ldots,2\ell\}$ as

$$\mathsf{T} := \{\ 2i - K_i \mid i \in [\ell],\ K_i \neq \perp\ \}. \tag{1}$$

See Fig. 1 for the illustrative example. Since an element in $\{1,2,\ldots,2\ell\}$ can be represented by a bit-string with length $\log 2\ell = O(\log\lambda)$ and T only consists of $O(\log\lambda)$ components, T can be represented by a bit-string with length $O(\log^2\lambda)$, which is much shorter than $\ell = \Theta(\lambda)$.

In the next step, we introduce a modified admissible hash function $\mathsf{F}_{\mathsf{MAH}}$ as

$$\mathsf{F}_{\mathsf{MAH}}(\mathsf{T},\mathsf{ID}) = \begin{cases} 0, & \text{if } \mathsf{T} \subseteq \mathsf{S}(\mathsf{ID}) \\ 1, & \text{otherwise} \end{cases} \qquad \text{where} \qquad \mathsf{S}(\mathsf{ID}) = \{\ 2i - C(\mathsf{ID})_i \mid i \in [\ell]\ \}.$$

Again, see Fig. 1 for the illustrative example. For T defined as above, we have

$$\mathsf{F}_{\mathsf{ADH}}(K,\mathsf{ID}) = \mathsf{F}_{\mathsf{MAH}}(\mathsf{T},\mathsf{ID}).$$

Namely, $\mathsf{F}_{\mathsf{ADH}}$ and $\mathsf{F}_{\mathsf{MAH}}$ are essentially the same functions, but they take different forms of inputs. The former takes K as the input, whereas the latter takes T, an encoded form of K, as the input. This fact suggests the possibility of the partitioning technique based on $\mathsf{F}_{\mathsf{MAH}}$, rather than $\mathsf{F}_{\mathsf{ADH}}$. Namely, we first choose $K \in \{0,1,\perp\}^\ell$ as specified, then set T as Eq.(1). The identity space is partitioned into two sets by $\mathsf{F}_{\mathsf{MAH}}(\mathsf{T},\cdot)$, which in turn is exactly the same partitioning made by $\mathsf{F}_{\mathsf{ADH}}(K,\cdot)$. Since the simulation strategy based on the function $\mathsf{F}_{\mathsf{MAH}}$ uses a much shorter secret randomness (i.e. T) than $\mathsf{F}_{\mathsf{ADH}}$, this opens up the possibility of constructing a much more compact IBE scheme.

Even given the above idea, the constructions of our IBE and VRF are not straightforward. Although the change is only in the encoding of the secret randomness, it might be the case that the construction of the function is incompatible with the underlying algebraic structures. In particular, $\mathsf{F}_{\mathsf{MAH}}$ seems to require more homomorphic capability than $\mathsf{F}_{\mathsf{ADH}}$. Indeed, even though we know how to construct IBE from bilinear maps using $\mathsf{F}_{\mathsf{ADH}}$ [BB04b], we do *not* know how to do it for $\mathsf{F}_{\mathsf{MAH}}$. In our lattice IBE, we can realize the idea by employing the fully key homomorphic technique introduced by [BGG$^+$14]. However, we have to be careful when applying the technique, otherwise we will end up with a super polynomial LWE as in [Yam16], which is undesirable both from the security and efficiency perspectives. For our VRF based on bilinear maps, we employ the fact that we can compute the function value by highly non-linear operations in the exponent.

2.2 Our First Lattice IBE

Our proposed IBE scheme follows the general framework for constructing a lattice IBE scheme [CHKP10, ABB10a, Yam16, ZCZ16] that associates to each identity ID the matrix $[\mathbf{A}\|\mathbf{B}_{\mathsf{ID}}] \in \mathbb{Z}_q^{n\times 2m}$. In the template construction, the main part of the ciphertext for ID contains $\mathbf{s}^\top[\mathbf{A}\|\mathbf{B}_{\mathsf{ID}}]+\mathbf{x}^\top$, where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{x}$ is a small noise term. On the other hand, a private key for ID is a short vector $\mathbf{e}$ satisfying $[\mathbf{A}\|\mathbf{B}_{\mathsf{ID}}]\mathbf{e} = \mathbf{u}$ for a random public vector $\mathbf{u}$.

We compute the matrix $\mathbf{B}_{\mathsf{ID}}$ using the fully key homomorphic technique of [BGG$^+$14]. Informally they showed that there exist algorithms $\mathsf{PubEval}$ and $\mathsf{TrapEval}$ that satisfy

$$\mathsf{PubEval}\left(\mathsf{F}, \{\mathbf{A}\mathbf{R}_i + y_i\mathbf{G}\}_{i\in[u]}\right) \;=\; \mathbf{A}\mathbf{R}_{\mathsf{F}} + \mathsf{F}(y)\cdot\mathbf{G} \qquad \text{where} \qquad \mathbf{R}_{\mathsf{F}} = \mathsf{TrapEval}\left(\mathsf{F}, \mathbf{A}, \{\mathbf{R}_i, y_i\}_{i\in[u]}\right).$$

Here, $\mathsf{F} : \{0,1\}^u \to \{0,1\}$ is some function, $\mathbf{R}_i$ is a matrix with small coefficients, and y_i is the i-th bit of the bit-string y. Furthermore, $\mathbf{R}_{\mathsf{F}}$ has small coefficients.

For our construction, we prepare random matrices $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_u$ in the master public key, where $u = \omega(\log^2 \lambda)$. Then, we set

$$\mathbf{B}_{\mathsf{ID}} = \mathsf{PubEval}(\ \mathsf{F}_{\mathsf{MAH}}(\ \cdot\ ,\mathsf{ID}),\ \{\mathbf{B}_i\}_{i\in[u]}\ ).$$

Here, we consider $\mathsf{F}_{\mathsf{MAH}}(\cdot, \mathsf{ID})$ as a function that takes an *binary string* representing T as an input. This is necessary to apply the result of [BGG$^+$14] without using the super-polynomial modulus. The security of the scheme is reduced to the LWE assumption, which says that given $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{w} \in \mathbb{Z}_q^m$, it is hard to distinguish whether $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_q^m$ or $\mathbf{w}^\top = \mathbf{s}^\top\mathbf{A} + \mathbf{x}'^\top$ for some noise term $\mathbf{x}'$. To prove security, we set the matrices $\{\mathbf{B}_i\}$ in the master public key as

$$\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + \mathsf{T}_i \cdot \mathbf{G}$$

where $\mathbf{A}$ is from the problem instance of the LWE, $\mathbf{R}_i$ is a random matrix with small coefficients, and $\mathsf{T}_i \in \{0,1\}$ is the i-th bit of the binary representation of T. Due to the leftover hash lemma, the master public key is correctly distributed. By the properties of $\mathsf{PubEval}$ and $\mathsf{TrapEval}$, we have

$$\mathbf{B}_{\mathsf{ID}} = \mathbf{A}\mathbf{R}_{\mathsf{ID}} + \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, \mathsf{ID})\cdot\mathbf{G} \qquad \text{where} \qquad \mathbf{R}_{\mathsf{ID}} = \mathsf{TrapEval}\left(\mathsf{F}_{\mathsf{MAH}}(\ \cdot\ ,\mathsf{ID}), \mathbf{A}, \{\mathbf{R}_i, \mathsf{T}_i\}_{i\in[u]}\right).$$

Furthermore, by the property of $\mathsf{F}_{\mathsf{MAH}}$, we have

$$\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, \mathsf{ID}^{(1)}) = \cdots = \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, \mathsf{ID}^{(Q)}) = 1 \ \wedge\ \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, \mathsf{ID}^{\star}) = 0 \tag{2}$$

with noticeable probability, where $\mathsf{ID}^\star$ is the challenge identity, and $\mathsf{ID}^{(1)}, \ldots, \mathsf{ID}^{(Q)}$ are identities for which the adversary has made key extraction queries. If this condition holds, the simulation will be successful. The key extraction queries for $\mathsf{ID} \in \{\mathsf{ID}^{(1)}, \ldots, \mathsf{ID}^{(Q)}\}$ can be handled by using $\mathbf{R}_{\mathsf{ID}}$ as a $\mathbf{G}$-trapdoor [MP12] for the matrix $[\mathbf{A} \| \mathbf{B}_{\mathsf{ID}}] = [\mathbf{A} \| \mathbf{A}\mathbf{R}_{\mathsf{ID}} + \mathbf{G}]$. The generation of the challenge ciphertext is also possible by computing

$$\mathbf{w}^{\top}[\mathbf{I} \| \mathbf{R}_{\mathsf{ID}^\star}] = \left(\mathbf{s}^{\top}\mathbf{A} + \mathbf{x}'^{\top}\right) \cdot [\mathbf{I} \| \mathbf{R}_{\mathsf{ID}^\star}] = \mathbf{s}^{\top}[\mathbf{A} \| \mathbf{B}_{\mathsf{ID}^\star}] + \underbrace{\mathbf{x}'^{\top}[\mathbf{I} \| \mathbf{R}_{\mathsf{ID}^\star}]}_{\text{noise term}}.$$

A subtle point here is that the noise term above is not correctly distributed. However, this problem can be resolved by the technique in [KY16] (Lemma 2).

Finally, we remark that our actual construction is different from the above in two points. First, we do not use the (general) fully key homomorphic algorithm of [BGG+14] to compute $\mathbf{B}_{\mathsf{ID}}$ and $\mathbf{R}_{\mathsf{ID}}$. If we use the algorithm in a naive way, the coefficients of $\mathbf{R}_{\mathsf{ID}}$ will become super-polynomial, which somewhat nullifies the merit of having smaller number of matrices. Instead, we show a direct algorithm to compute $\mathbf{B}_{\mathsf{ID}}$ and $\mathbf{R}_{\mathsf{ID}}$ using the technique of [GV15], such that the coefficients of $\mathbf{R}_{\mathsf{ID}}$ are polynomially bounded. The second difference is that we add a matrix $\mathbf{B}_0$ to the master public key and use the matrix $[\mathbf{A} \| \mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}]$ in the encryption and the key generation, instead of $[\mathbf{A} \| \mathbf{B}_{\mathsf{ID}}]$. This change is introduced because of a subtle technical reason to make the security proof easier.

2.3 Our First VRF

Our VRF is constructed on bilinear maps and obtained by incorporating our technique with the previous inversion-based VRF schemes [DY05, BMR10]. In the scheme, we set the function as

$$\mathsf{V}_{\mathsf{sk}}(X) = e(g, h)^{1/\theta_X}, \tag{3}$$

where the value $\theta_X = \mathbb{Z}_p^*$ is deterministically computed by the input X. Let us ignore the problem of how we add the verifiability to the scheme for the time being and start with the overview of the security proof for the scheme as a (plain) PRF. The security will be proven under the L-DDH assumption, which says that given $(h, \hat{g}, \hat{g}^{\alpha}, \ldots \hat{g}^{\alpha^L}, \Psi)$, it is infeasible to distinguish whether $\Psi \xleftarrow{\$} \mathbb{G}_T$ or $\Psi = e(\hat{g}, h)^{1/\alpha}$. As before, we sample T and partition the input space into two sets by $\mathsf{F}_{\mathsf{MAH}}$. By the property and definition of $\mathsf{F}_{\mathsf{MAH}}$, we have

$$\mathsf{T} \not\subseteq \mathsf{S}(X^{(1)}) \ \wedge \ \cdots \ \wedge \ \mathsf{T} \not\subseteq \mathsf{S}(X^{(Q)}) \ \wedge \ \mathsf{T} \subseteq \mathsf{S}(X^{\star})$$

with noticeable probability, where $X^\star$ is the challenge input and $X^{(1)}, \ldots, X^{(Q)}$ are the inputs for which the adversary has made evaluation queries. Our strategy to prove the security is to embed the problem instance and T into the parameters of the scheme so that we have

$$\theta_X = \mathsf{P}_X(\alpha) \qquad \text{and} \qquad g = \hat{g}^{\mathsf{Q}(\alpha)}.$$

Here, $\mathsf{P}_X(\mathsf{Z})$ is a polynomial in $\mathbb{Z}_p[\mathsf{Z}]$ that depends on X and $\mathsf{Q}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ is some fixed polynomial. We want $\mathsf{P}_X(\mathsf{Z})$ and $\mathsf{Q}(\mathsf{Z})$ to satisfy the following property: There exist $\xi_X \in \mathbb{Z}_p^*$ and $\mathsf{R}_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that

$$\frac{\mathsf{Q}(\mathsf{Z})}{\mathsf{P}_X(\mathsf{Z})} = \begin{cases} \dfrac{\xi_X}{\mathsf{Z}} + \mathsf{R}_X(\mathsf{Z}) & \text{if} \quad \mathsf{T} \subseteq \mathsf{S}(X) \\ \mathsf{R}_X(\mathsf{Z}) & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S}(X) \end{cases}. \tag{4}$$

If the above holds, the simulation will be successful. To answer the evaluation query on input $X \in \{X^{(1)}, \ldots, X^{(Q)}\}$, we compute $e(\hat{g}^{\mathsf{R}_X(\alpha)}, h)$. This is a valid answer, since we have $\mathsf{T} \not\subseteq \mathsf{S}(X)$ and thus

$$e(\hat{g}^{\mathsf{R}_X(\alpha)}, h) = e(\hat{g}^{\mathsf{Q}(\alpha)/\mathsf{P}_X(\alpha)}, h) = e(g^{1/\mathsf{P}_X(\alpha)}, h) = e(g, h)^{1/\theta_X}.$$

To answer the challenge query, we compute $\Psi^{\xi_{X^\star}} \cdot e\left(\hat{g}^{\mathsf{R}_{X^\star}(\alpha)}, h\right)$. If $\Psi \xleftarrow{\$} \mathbb{G}_T$, it is a random element in $\mathbb{G}_T$, as desired. On the other hand, if $\Psi = e(\hat{g}, h)^{1/\alpha}$, we have

$$\Psi^{\xi_{X^\star}} \cdot e\left(\hat{g}^{\mathsf{R}_{X^\star}(\alpha)}, h\right) = e\left(\hat{g}^{\mathsf{Q}(\alpha)/\mathsf{P}_{X^\star}(\alpha)}, h\right) = e\left(g^{1/\mathsf{P}_{X^\star}(\alpha)}, h\right) = e(g, h)^{1/\theta_{X^\star}}$$

which is the correct value. Now we have to find the polynomials with the desired property (namely, Eq. (4)). Let us take $\mathsf{P}_X(\mathsf{Z})$ to be the following form:[‡]

$$\mathsf{P}_X(\mathsf{Z}) = \prod_{i \in [\eta], j \in [\ell]} (\mathsf{Z} - t_i + s_j) \qquad \text{where} \qquad \mathsf{T} = \{t_1, \ldots, t_\eta\} \quad \text{and} \quad \mathsf{S}(X) = \{s_1, \ldots, s_\ell\}.$$

In some sense, $\mathsf{P}_X(\mathsf{Z})$ checks $(t_i \overset{?}{=} s_j)$ in a brute-force manner. We can see that $\mathsf{P}_X(\mathsf{Z})$ can be divided by Z exactly $|\mathsf{T} \cap \mathsf{S}(X)|$ times. Furthermore, we have $|\mathsf{T} \cap \mathsf{S}(X)| = |\mathsf{T}| = \eta \Leftrightarrow \mathsf{T} \subseteq \mathsf{S}(X)$. This motivates us to define $\mathsf{Q}(\mathsf{Z})$ as follows:

$$\mathsf{Q}(\mathsf{Z}) = \mathsf{Z}^{\eta - 1} \cdot \prod_{a \neq 0} (\mathsf{Z} + a), \tag{5}$$

where the product is taken for sufficiently many $a \neq 0$, so that the latter part of $\mathsf{Q}(\mathsf{Z})$ can be divided by any factor of $\mathsf{P}_X(\mathsf{Z})$ except for Z. It is easy to see that $\mathsf{Q}(\mathsf{Z})$ can be divided by Z exactly $\eta - 1$ times. These imply that $\mathsf{Q}(\mathsf{Z})$ can be divided by $\mathsf{P}_X(\mathsf{Z})$, if and only if the multiplicity of Z in $\mathsf{P}_X(\mathsf{Z})$ is at most $\eta - 1$. This fact allows us to prove Eq.(4).

Finally, we go back and see how our actual construction works. We set the verification key as $\mathsf{vk} = (g, h, \{W_i = g^{w_i}\}_{i \in [\eta]})$ and choose θ_X as

$$\theta_X = \prod_{(i,j) \in [\eta] \times [\ell]} \underbrace{(w_i + s_j)}_{:= \theta_{i,j}} = \prod_{i \in [\eta]} \underbrace{\left(\prod_{j \in [\ell]} (w_i + s_j) \right)}_{\phi_i} \tag{6}$$

and set the function value as $\mathsf{V}_{\mathsf{sk}}(X) = e(g, h)^{1/\theta_X}$. The form of θ_X reflects the "brute-force structure" that has appeared in $\mathsf{P}_X(\mathsf{Z})$. To generate a proof for the function value, we take the "step ladder approach" [Lys02, ACF09, HW10]. Namely, we publish values of the form $g^{1/\theta_{1,1}}, g^{1/\theta_{1,1}\theta_{1,2}}, \ldots, g^{1/\theta_{1,1} \cdots \theta_{\eta,\ell}} = g^{1/\theta_X}$. The correctness of the function value can be verified by the pairing computations using these terms. While this scheme achieves very short verification key, the proofs for the function values are very long. We can make the proofs much shorter by a simple trick. We introduce additional helper components $\{g^{w_i^j}\}_{(i,j) \in [\eta] \times [\ell]}$ to the verification key. Instead of publishing the proof above, we publish $g^{1/\phi_1}, g^{1/\phi_1 \phi_2}, \ldots, g^{1/\phi_1 \cdots \phi_\eta} = g^{1/\theta_X}$ as a proof. Thanks to the helper components, we can verify whether the function value is correct using the proof.

[‡] For simplicity, we use a polynomial that is slightly different from the actual proof.

2.4 Other Constructions

Partitioning with Yet Another Function. We propose another function $\mathsf{F}_{\mathsf{AFF}}$, which is also useful to perform the partitioning technique. The main advantage of the function over $\mathsf{F}_{\mathsf{MAH}}$ is that it achieves even shorter secret randomness K of length $\omega(\log \lambda)$. Here, we begin by reviewing $\mathsf{F}_{\mathsf{WAT}}$, a slight variant of the celebrated Waters' hash [Wat05], and then gradually modify it to our $\mathsf{F}_{\mathsf{AFF}}$. Let the identity space of IBE (or input space of VRF) be $\{0,1\}^k$. The function $\mathsf{F}_{\mathsf{WAT}}$ is defined as

$$\mathsf{F}_{\mathsf{WAT}}(\ K = (\{\alpha_i\}_{i\in[k]}, \beta), \mathsf{ID}\ ) = \begin{cases} 0, & \text{if} \quad (\sum_{i\in[k]} \alpha_i \mathsf{ID}_i) + \beta = 0 \\ 1, & \text{otherwise} \end{cases} \qquad \text{where} \quad \alpha_i, \beta \in \mathbb{Z},\ \mathsf{ID} \in \{0,1\}^k$$

Here, ID_i is the i-th bit of ID. In order for the function to be useful, we should choose the random secret K so that

$$\Pr_K \left[\ \mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{(1)}) = 1\ \wedge\ \cdots\ \wedge\ \mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{(Q)}) = 1\ \wedge\ \mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{\star}) = 0\ \right]$$

is noticeable. By a standard analysis, one can show that it suffices to satisfy the following two requirements:

(A) $\Pr_K[\ \mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{\star}) = 0\ ]$ is noticeable.

(B) $\Pr_K[\ \mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{(i)}) = 0 \mid \mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{\star}) = 0\ ]$ is sufficiently small for all $i \in [Q]$.

In order to satisfy the requirements, one way to choose is $\alpha_1, \ldots, \alpha_k \overset{\$}{\leftarrow} [1, 4Q]$ and $\beta \overset{\$}{\leftarrow} [-4kQ, 0]$. As for requirement (A), we have

$$\Pr_K[\mathsf{F}_{\mathsf{WAT}}(K, \mathsf{ID}^{\star}) = 0] = \Pr_{\alpha,\beta}\left[\beta = -\sum_{i\in[k]} \alpha_i \mathsf{ID}^{\star}_i\right] = \frac{1}{4kQ+1}$$

where the second equality follows from $-4kQ \le \sum_{i\in[k]} \alpha_i \mathsf{ID}^{\star}_i \le 0$. We can see that the probability is noticeable as desired. The main observation here is that since the value of each α_i is polynomially bounded and $\mathsf{ID}^{\star}_i \in \{0,1\}$, the total sum is also confined within the polynomially bounded range and thus can be guessed with noticeable probability. Requirement (B) can be proven by exploiting a certain kind of pairwise independence of $\mathsf{F}_{\mathsf{WAT}}(K, \cdot)$.

The problem of the above function is that it requires long secret randomness K, whose length is linear in k. As the first attempt to shorten this, we could consider a modified function $\mathsf{F}'_{\mathsf{WAT}}$ defined as

$$\mathsf{F}'_{\mathsf{WAT}}(\ K = (\alpha, \beta),\ \mathsf{ID}\ ) = \begin{cases} 0, & \text{if} \quad \alpha\mathsf{ID} + \beta = 0 \\ 1, & \text{otherwise} \end{cases} \qquad \text{where} \quad \alpha, \beta \in \mathbb{Z}, \mathsf{ID} \in [2^k - 1]$$

where we interpret $\mathsf{ID} \in \{0,1\}^k$ as an integer in $[2^k - 1]$ by the natural bijection. While it is easy to satisfy requirement (B), we no longer know how to satisfy requirement (A) at the same time. Even if the size of α is polynomially bounded, $\alpha \cdot \mathsf{ID}$ can be very large, and we can not guess the value better than with exponentially small probability.

To resolve the problem, we further modify the function and obtain our final function $\mathsf{F}_{\mathsf{AFF}}$ defined as follows:

$$\mathsf{F}_{\mathsf{AFF}}(\ K = (\alpha, \beta, \rho),\ \mathsf{ID}\ ) = \begin{cases} 0, & \text{if} \quad \alpha\mathsf{ID} + \beta \equiv 0 \mod \rho \\ 1, & \text{otherwise} \end{cases} \qquad \text{where} \quad \alpha, \beta, \rho \in \mathbb{Z},\ \mathsf{ID} \in [2^k - 1].$$

Here, we choose ρ to be a random polynomial-size prime. Now, we can satisfy requirement (A), since we only have to guess $(\alpha \cdot \mathsf{ID} \mod \rho)$, for which there are only a polynomial number of candidates. However, making the size of ρ polynomial causes a subtle problem regarding requirement (B). Let us consider the case where an adversary makes queries such that $\mathsf{ID}^\star = \mathsf{ID}^{(1)} + \rho$. In such a case, we have $\mathsf{F}_{\mathsf{AFF}}(K, \mathsf{ID}^\star) = \mathsf{F}_{\mathsf{AFF}}(K, \mathsf{ID}^{(1)})$ and the simulation fails with probability 1, no matter how we choose α and β. Such queries can be made with noticeable probability, since ρ is polynomial-size and the adversary can guess the value with noticeable probability. However a small subtlety is that the probability does not need to be negligible in order to satisfy requirement (B). Due to this observation, by choosing ρ randomly from a large enough domain (concretely, from $[kQ^2/\epsilon, 4kQ^2/\epsilon]$), we can make the probability of such queries being made sufficiently small, hence satisfying requirement (A) and (B).

New IBE and VRF Based on the Function. Based on the function $\mathsf{F}_{\mathsf{AFF}}$, we propose a lattice based IBE scheme and a VRF scheme on bilinear groups. To construct a lattice based IBE scheme, we follow the same template as the case of $\mathsf{F}_{\mathsf{MAH}}$ and set $\mathbf{B}_{\mathsf{ID}} = \mathsf{PubEval}(\mathsf{F}_{\mathsf{AFF}}(\ \cdot\ , \mathsf{ID}), \{\mathbf{B}_i\}_{i \in [u]})$. Again, if we use the fully key homomorphic algorithm of [BGG+14] naively, the scheme will require super polynomial modulus q. To avoid this, to compute $\mathbf{B}_{\mathsf{ID}}$, we first compute a description of a log-depth circuit corresponding to $\mathsf{F}_{\mathsf{AFF}}$. Such a circuit exists by the classical result of Beam, Cook, and Hoover [BCH86], who showed that the computation of division can be performed in $\mathbf{NC}^1$, since division implies modulo ρ arithmetic. Then, we convert the log-depth circuit into a branching program using the Barrington's theorem [Bar89]. Finally, we use the key homomorphic algorithm for branching programs in [GV15]. Note that similar approach was also taken in [BL16] to homomorphically evaluate a PRF. To construct a VRF based on bilinear groups, we again take advantage of the fact that $\mathsf{F}_{\mathsf{AFF}}$ can be computed by a log-depth circuit. This fact is necessary for our VRF to be proven secure under a polynomial-size assumption, since our security proof requires 2^d-DDH assumption, where d is the depth of the circuit.

3 Preliminaries

Notation. We denote by $[a]$ a set $\{1, 2, \ldots, a\}$ for any integer $a \in \mathbb{N}$. For a set S, $|S|$ denotes its size. We treat a vector as a column vector. If $\mathbf{A}_1$ is an $n \times m$ and $\mathbf{A}_2$ is an $n \times m'$ matrix, then $[\mathbf{A}_1 \| \mathbf{A}_2]$ denotes the $n \times (m + m')$ matrix formed by concatenating $\mathbf{A}_1$ and $\mathbf{A}_2$. We use similar notation for vectors. For a vector $\mathbf{u} \in \mathbb{Z}^n$, $\|\mathbf{u}\|$ and $\|\mathbf{u}\|_\infty$ denote its ℓ_2 and ℓ_∞ norm respectively. Similarly, for a matrix $\mathbf{R}$, $\|\mathbf{R}\|_\infty$ denotes its infinity norm. $\|\mathbf{R}\|_2$ is the operator norm of $\mathbf{R}$. Namely, $\|\mathbf{R}\|_2 := \sup_{\|\mathbf{x}\|=1} \|\mathbf{Rx}\|$. For a function $f(\cdot) : \mathbb{N} \to \mathbb{R}_{\geq 0}$, we say that the function is negligible when for every polynomial $g(\cdot)$ and all sufficiently large λ we have $f(\lambda) < |1/g(\lambda)|$. We say that the function is noticeable when there exists a polynomial $g(\cdot)$ such that we have $f(\lambda) \geq |1/g(\lambda)|$ for all λ.

3.1 Identity-Based Encryption

Let $\{0,1\}^k$ be the identity space of the scheme. An IBE scheme is defined by the following four algorithms.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$: The setup algorithm takes as input a security parameter 1^λ and outputs a master public key mpk and a master secret key msk.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID}) \to \mathsf{sk}_{\mathsf{ID}}$: The key generation algorithm takes as input the master public key mpk, the master secret key msk, and an identity $\mathsf{ID} \in \{0,1\}^k$. It outputs a private key $\mathsf{sk}_{\mathsf{ID}}$. We assume that ID is implicitly included in $\mathsf{sk}_{\mathsf{ID}}$.

$\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M}) \to \mathsf{ct}$: The encryption algorithm takes as input a master public key mpk, an identity $\mathsf{ID} \in \{0,1\}^k$, and a message M, It outputs a ciphertext ct.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{ct}) \to \mathsf{M}$ or $\perp$: The decryption algorithm takes as input the master public key mpk, a private key $\mathsf{sk}_{\mathsf{ID}}$, and a ciphertext ct. It outputs the message M or $\perp$, which means that the ciphertext is not in a valid form.

Definition 1. *We say that a tuple of efficient algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ *is an adaptively-anonymous IBE, if all the following properties hold.*

Correctness. *We require correctness of decryption: that is, for all* λ*, all* $\mathsf{ID} \in \{0,1\}^k$*, and all* M *in the specified message space,* $\Pr[\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})) = \mathsf{M}] = 1 - \mathsf{negl}(n)$ *holds, where the probability is taken over the randomness of the algorithms.*

Adaptive Anonymous Security. *This security notion is defined by the following game between a challenger and an adversary* $\mathcal{A}$*.*

Setup. *At the outset of the game, the challenger runs* $\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$ *and gives* mpk *to* $\mathcal{A}$*.*

Phase 1. $\mathcal{A}$ *may adaptively make key-extraction queries. If* $\mathcal{A}$ *submits* $\mathsf{ID} \in \{0,1\}^k$ *to the challenger, the challenger returns* $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$*.*

Challenge Phase. *At some point,* $\mathcal{A}$ *outputs a message* M *and an identity* $\mathsf{ID}^\star \in \{0,1\}^k$*, on which it wishes to be challenged. Then, the challenger picks a random coin* $\mathsf{coin} \overset{\$}{\leftarrow} \{0,1\}$ *and a random ciphertext* $\mathsf{ct}_1^\star \overset{\$}{\leftarrow} \mathcal{C}$ *from the ciphertext space. If* $\mathsf{coin} = 0$*, it runs* $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}^\star, \mathsf{M}) \to \mathsf{ct}_0^\star$ *and gives the challenge ciphertext* $\mathsf{ct}_0^\star$ *to* $\mathcal{A}$*. If* $\mathsf{coin} = 1$*, it gives* $\mathsf{ct}_1^\star$ *to* $\mathcal{A}$*.*

Phase 2. *After the challenge query,* $\mathcal{A}$ *may continue to make key-extraction queries, with the added restriction that* $\mathsf{ID} \neq \mathsf{ID}^\star$*.*

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

The advantage of $\mathcal{A}$ *is defined as* $|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}|$*. We say that the scheme satisfies adaptively-anonymous security if the advantage of any PPT* $\mathcal{A}$ *is negligible.*

3.2 Verifiable Random Function

A verifiable random function consists of the following three algorithms.

$\mathsf{Gen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$: The generation algorithm takes as input a security parameter 1^λ and outputs a verification key vk and a secret key sk.

$\mathsf{Eval}(\mathsf{sk}, X) \to (Y, \pi)$: The evaluation algorithm takes as input the secret key sk and an input $X \in \{0,1\}^k$ and outputs a function value $Y \in \mathcal{Y}$, where $\mathcal{Y}$ is a finite set, and a proof π.

$\mathsf{Verify}(\mathsf{vk}, X, Y, \pi) \to 1$ or 0: The verification algorithm takes as input vk, $X \in \{0,1\}^k$, $Y \in \mathcal{Y}$, and a proof π, and outputs a bit.

Definition 2. *We say that a tuple of algorithms* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Verify})$ *is a verifiable random function (VRF), if all the following properties hold.*

Correctness. *Algorithm* $\mathsf{Gen}, \mathsf{Eval}, \mathsf{Verify}$ *are polynomial-time algorithms, and for all* $(\mathsf{vk}, \mathsf{sk}) \overset{\$}{\leftarrow} \mathsf{Gen}(1^\lambda)$ *and all* $X \in \{0,1\}^k$ *holds: if* $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$*, then* $\mathsf{Verify}(\mathsf{vk}, X, Y, \pi) = 1$*.*

Unique provability. *For all strings* $\mathsf{vk} \in \{0,1\}^*$ *(not necessarily generated by* Gen*) and all* $X \in \{0,1\}^k$*, there does not exist any* (Y_0, π_0, Y_1, π_1) *such that* $Y_0 \neq Y_1$ *and* $\mathsf{Verify}(\mathsf{vk}, X, Y_0, \pi_0) = \mathsf{Verify}(\mathsf{vk}, X, Y_1, \pi_1) = 1$*.*

Pseudorandomness. *This security notion is defined by the following game between a challenger and an adversary $\mathcal{A}$.*

Setup. *At the outset of the game, the challenger runs* $\mathsf{Gen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$ *and gives* vk *to* $\mathcal{A}$.

Phase 1. *$\mathcal{A}$ may adaptively query the evaluation of the function. If $\mathcal{A}$ submits $X \in \{0,1\}^k$ to the challenger, the challenger returns* $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$.

Challenge Query. *At some point, $\mathcal{A}$ makes the challenge query. If $\mathcal{A}$ outputs $X^\star \in \{0,1\}^k$, the challenger picks random coin* $\mathsf{coin} \xleftarrow{\$} \{0,1\}$. *Then it runs* $(Y_0^\star, \pi_0^\star) \leftarrow \mathsf{Eval}(\mathsf{sk}, X^\star)$ *and picks* $Y_1^\star \xleftarrow{\$} \mathcal{Y}$. *Finally it returns* $Y^\star_{\mathsf{coin}}$ *to* $\mathcal{A}$.

Phase 2. *After the challenge query, $\mathcal{A}$ may continue querying the evaluation of the function with the added restriction that $X \neq X^\star$. The challenger returns* $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$.

Guess. *Finally, $\mathcal{A}$ outputs guess* $\widehat{\mathsf{coin}}$ *for* coin.

The advantage of $\mathcal{A}$ is defined as $|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}|$. *We say that the scheme satisfies pseudorandomness if the advantage of any PPT $\mathcal{A}$ is negligible.*

3.3 Preliminaries on Lattices

Gaussian Distributions. For an integer $m > 0$, let $D_{\mathbb{Z}^m,\sigma}$ be the discrete Gaussian distribution over $\mathbb{Z}^m$ with parameter $\sigma > 0$. Regarding the Gaussian distributions, the following lemmas hold.

Lemma 1 ([Reg05], Lemma 2.5)**.** *We have* $\Pr[\ \|\mathbf{x}\| > \sigma\sqrt{m}\ :\ \mathbf{x} \xleftarrow{\$} D_{\mathbb{Z}^m,\sigma}\ ] \leq 2^{-2m}$.

Lemma 2 ([KY16], Lemma 1)**.** *Let q, m, m' be positive integers and r a positive real satisfying $r > \max\{\omega(\sqrt{\log m}), \omega(\sqrt{\log m'})\}$. Let $\mathbf{b} \in \mathbb{Z}_q^m$ be arbitrary and $\mathbf{x}$ chosen from $D_{\mathbb{Z}^m,r}$. Then for any $\mathbf{V} \in \mathbb{Z}^{m \times m'}$ and positive real $s > \|\mathbf{V}\|_2$, there exists a PPT algorithm* $\mathsf{ReRand}(\mathbf{V}, \mathbf{b} + \mathbf{x}, r, s)$ *that outputs $\mathbf{b}'$ such that $\mathbf{b}'^\top = \mathbf{b}^\top \mathbf{V} + \mathbf{x}'^\top \in \mathbb{Z}_q^{m' \times 1}$ where $\mathbf{x}'$ is distributed statistically close to $D_{\mathbb{Z}^{m'},2rs}$.*

Learning with Errors (LWE) Assumption. We define the learning with errors (LWE) problem, which was introduced by Regev [Reg05].

Definition 3 (LWE)**.** *For an integers $n = n(\lambda)$, $m = m(n)$, a prime integer $q = q(n) > 2$, a real number $\alpha \in (0,1)$, and a PPT algorithm $\mathcal{A}$, an advantage for the learning with errors problem* $\mathsf{dLWE}_{n,m,q,\alpha}$ *of $\mathcal{A}$ is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\alpha}} = \left|\Pr\left[\mathcal{A}(\mathbf{A}, \mathbf{s}^\top\mathbf{A} + \mathbf{x}^\top) \to 1\right] - \Pr\left[\mathcal{A}(\mathbf{A}, \mathbf{w}^\top + \mathbf{x}^\top) \to 1\right]\right|$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{x} \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha q}$, $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_q^m$. We say that $\mathsf{dLWE}_{n,m,q,\alpha}$ *assumption holds if* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\alpha}}$ *is negligible for all PPT $\mathcal{A}$.*

Regev [Reg05] (see also [GKV10]) showed that solving $\mathsf{dLWE}_{n,m,q,\alpha}$ for $\alpha q > 2\sqrt{2n}$ is (quantumly) as hard as approximating the 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 Regev's reduction were achieved [Pei09, BLP$^+$13].

Gadget Matrix. Let $m > n\lceil \log q \rceil$. There is a fixed full-rank matrix $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ such that there exists a deterministic polynomial-time algorithm $\mathbf{G}^{-1}$ which takes the input $\mathbf{U} \in \mathbb{Z}_q^{n \times m}$ and outputs $\mathbf{V} = \mathbf{G}^{-1}(\mathbf{U})$ such that $\mathbf{V} \in \{0,1\}^{m \times m}$ and $\mathbf{G}\mathbf{V} = \mathbf{U}$.

Trapdoors. Here, we follow the presentation of [BV16]. Let $n, m, q \in \mathbb{N}$ and consider a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$. For all $\mathbf{V} \in \mathbb{Z}_q^{n\times m'}$, we let $\mathbf{A}_\sigma^{-1}(\mathbf{V})$ be a distribution that is a Gaussian $(D_{\mathbb{Z}^m,\sigma})^{m'}$ conditioned on $\mathbf{A} \cdot \mathbf{A}_\sigma^{-1}(\mathbf{V}) = \mathbf{V}$. A σ-trapdoor for $\mathbf{A}$ is a procedure that can sample from the distribution $\mathbf{A}_\sigma^{-1}(\mathbf{V})$ in time $\mathsf{poly}(n, m, m', \log q)$, for any $\mathbf{V}$. We slightly overload notation and denote a σ-trapdoor for $\mathbf{A}$ by $\mathbf{A}_\sigma^{-1}$. We have the following:

Lemma 3 (Properties of Trapdoors [GPV08, ABB10a, CHKP10, ABB10b, MP12, BLP+13])**.** *Lattice trapdoors exhibit the following properties.*

1. *Given $\mathbf{A}_\sigma^{-1}$, one can obtain $\mathbf{A}_{\sigma'}^{-1}$ for any $\sigma' \geq \sigma$.*
2. *Given $\mathbf{A}_\sigma^{-1}$, one can obtain $[\mathbf{A}\|\mathbf{B}]_\sigma^{-1}$ and $[\mathbf{B}\|\mathbf{A}]_\sigma^{-1}$ for any $\mathbf{B}$.*
3. *For all $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{R} \in \mathbb{Z}^{m\times m}$, with $m \geq n\lceil \log q\rceil$, one can obtain $[\mathbf{AR} + \mathbf{G}\|\mathbf{A}]_\sigma^{-1}$ for $\sigma = m \cdot \|\mathbf{R}\|_\infty \cdot \omega(\sqrt{\log m})$.*
4. *There exists an efficient procedure $\mathsf{TrapGen}(1^n, 1^m, q)$ that outputs $(\mathbf{A}, \mathbf{A}_{\sigma_0}^{-1})$ where $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ for some $m = O(n \log q)$ and is 2^{-n}-close to uniform, where $\sigma_0 = \omega(\sqrt{n \log q \log m})$.*
5. *For $\mathbf{A}_\sigma^{-1}$ and $\mathbf{u} \in \mathbb{Z}_q^n$, it follows $\Pr[\ \|\mathbf{A}_\sigma^{-1}(\mathbf{u})\| > \sqrt{m}\sigma\ ] = \mathsf{negl}(n)$.*

3.4 Preliminaries on Bilinear Maps

Certified Bilinear Group Generators. We define certified bilinear group generators following [HJ16]. We require that there is an efficient bilinear group generator algorithm GrpGen that on input 1^λ and outputs a description Π of bilinear groups $\mathbb{G}, \mathbb{G}_T$ with prime order p and a map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We also require that GrpGen is certified, in the sense that there is an efficient algorithm GrpVfy that on input a (possibly incorrectly generated) description of the bilinear groups and outputs whether the description is valid or not. Furthermore, we require that each group element has unique encoding, which can be efficiently recognized.

Definition 4. *A bilinear group generator is a probabilistic polynomial-time algorithm* GrpGen *that takes as input a security parameter λ (in unary) and outputs $\Pi = (p, \mathbb{G}, \mathbb{G}_T, \circ, \circ_T, e, \phi(1)) \overset{\$}{\leftarrow}$* $\mathsf{GrpGen}(1^\lambda)$ *such that the following requirements are satisfied.*

1. *p is prime and $\log(p) = \Omega(\lambda)$.*
2. *$\mathbb{G}$ and $\mathbb{G}_T$ are subsets of $\{0, 1\}^*$, defined by algorithmic descriptions of maps $\phi : \mathbb{Z}_p \to \mathbb{G}$ and $\phi_T : \mathbb{Z}_p \to \mathbb{G}_T$.*
3. *$\circ$ and $\circ_T$ are algorithmic descriptions of efficiently computable (in the security parameter) maps $\circ : \mathbb{G} \times \mathbb{G} \to \mathbb{G}$ and $\circ_T : \mathbb{G}_T \times \mathbb{G}_T \to \mathbb{G}_T$, such that*
   - *$(\mathbb{G}, \circ)$ and $(\mathbb{G}_T, \circ_T)$ form algebraic groups,*
   - *ϕ is a group isomorphism form $(\mathbb{Z}_p, +)$ to $(\mathbb{G}, \circ)$, and*
   - *ϕ_T is a group isomorphism from $(\mathbb{Z}_p, +)$ to $(\mathbb{G}_T, \circ_T)$.*
4. *e is an algorithmic description of an efficiently computable (in the security parameter) bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We require that e is non-degenerate, that is,*

$$x \neq 0 \to e(\phi(x), \phi(x)) \neq \phi_T(0).$$

Definition 5. *We say that group generator* GrpGen *is certified, if there exists a deterministic polynomial-time algorithm* GrpVfy *with the following properties.*

Parameter validation. *Given a string* Π *(which is not necessarily generated by* GrpGen*), algorithm* $\mathsf{GrpVfy}(\Pi)$ *outputs* 1 *if and only if* Π *has the form*

$$\Pi = (p, \mathbb{G}, \mathbb{G}_T, \circ, \circ_T, e, \phi(1))$$

and all requirements from Definition 4 are satisfied.

Recognition and unique representation of elements of $\mathbb{G}$. *Furthermore, we require that each element in* $\mathbb{G}$ *has a unique representation, which can be efficiently recognized. That is, on input two strings* Π *and* s*,* $\mathsf{GrpVfy}(\Pi, s)$ *outputs* 1 *if and only if* $\mathsf{GrpVfy}(\Pi) = 1$ *and it holds that* $s = \phi(x)$ *for some* $x \in \mathbb{Z}_p$*. Here* $\phi : \mathbb{Z}_p \to \mathbb{G}$ *denotes the fixed group isomorphism contained in* Π *to specify the representation of elements of* $\mathbb{G}$ *(see Definition 4).*

L-Diffie-Hellman Assumptions.

Definition 6 (L-Diffie-Hellman Assumptions)**.** *For a PPT algorithm* $\mathcal{A}$*, an advantage for the decisional* L*-Diffie Hellman problem* L-DDH *of* $\mathcal{A}$ *with respect to* GrpGen *is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A}}^{L\text{-}\mathsf{DDH}} = |\Pr[\mathcal{A}(\Pi, \hat{g}, h, \hat{g}^{\alpha}, \hat{g}^{\alpha^2}, \dots \hat{g}^{\alpha^L}, \Psi_0) \to 1] - \Pr[\mathcal{A}(\Pi, \hat{g}, h, \hat{g}^{\alpha}, \hat{g}^{\alpha^2}, \dots \hat{g}^{\alpha^L}, \Psi_1) \to 1]|$$

where $\Pi \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$*,* $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$*,* $\hat{g}, h \xleftarrow{\$} \mathbb{G}$*,* $\Psi_0 = e(\hat{g}, h)^{1/\alpha}$*, and* $\Psi_1 \xleftarrow{\$} \mathbb{G}_T$*. We say that* L-DDH *assumption holds if* $\mathsf{Adv}_{\mathcal{A}}^{L\text{-}\mathsf{DDH}}$ *is negligible for all PPT* $\mathcal{A}$*.*

3.5 Known Facts

As observed in [Reg05, ABB10a], the following lemma is obtained as a corollary to the (general) leftover hash lemma.

Lemma 4. (Leftover Hash Lemma.) Let $q \in \mathbb{N}$ be an odd prime and let $m > (n+1)\log q + \omega(\log n)$. Let $\mathbf{R} \xleftarrow{\$} \{-1, 1\}^{m \times m}$ and $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ be uniformly random matrices. Then the distribution of $(\mathbf{A}, \mathbf{AR})$ is $\mathsf{negl}(n)$-close to the distribution of $(\mathbf{A}, \mathbf{A}')$.

Lemma 5. (Number of Primes.) For $a, b \in \mathbb{N}$ with $a \le b$, let $\pi(a, b)$ be the number of primes in $[a, b]$. If $a \ge 2^{21}$, we have

$$\frac{a}{\log_2 a} \le \pi(a+1, 4a)$$

Proof. In Lemma 5.3 of [BHJ+13], it is shown that $a/\log a \le \pi(1, a) \le 2a/\log a$ holds for $a \ge 2^{21}$. Therefore, we have

$$\pi(a+1, 4a) = \pi(1, 4a) - \pi(1, a) \ge \frac{4a}{\log 4a} - \frac{2a}{\log a} \ge \frac{3a}{\log a} - \frac{2a}{\log a} = \frac{a}{\log a}.$$

This completes the proof of Lemma 5. □

The following lemma is taken from [KY16] (see also Lemma 28 in the full version of [ABB10a]), and is implicit in [BR09, Jag15, Yam16].

Lemma 6 (Lemma 8 in [KY16], See also Lemma 28 in [ABB10a])**.** *Let us consider an IBE (resp. VRF) scheme and an adversary $\mathcal{A}$ that breaks the adaptively-anonymous security (resp. pseudorandomness) with advantage ϵ. Let the identity space (resp. input space) be $\mathcal{X}$ and consider a map γ that maps a sequence of elements in $\mathcal{X}$ to a value in $[0,1]$. We consider the following experiment. We first execute the security game for $\mathcal{A}$. Let $X^\star$ be the challenge identity (resp. challenge input) and $X_1, \ldots, X_Q$ be the identities (resp. inputs) for which key extraction queries (resp. evaluation queries) were made. We denote $\mathbb{X} = (X^\star, X_1, \ldots, X_Q)$. At the end of the game, we set $\mathsf{coin}' \in \{0,1\}$ as $\mathsf{coin}' = \widehat{\mathsf{coin}}$ with probability $\gamma(\mathbb{X})$ and $\mathsf{coin}' \overset{\$}{\leftarrow} \{0,1\}$ with probability $1 - \gamma(\mathbb{X})$. Then, the following holds.*

$$\left| \Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2} \right| \geq \gamma_{\min} \cdot \epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2}$$

where $\gamma_{\min}$ and $\gamma_{\max}$ are the maximum and the minimum of $\gamma(\mathbb{X})$ taken over all possible $\mathbb{X}$, respectively.

Though the lemma was proven only for IBE in [KY16], the same proof works also for VRF.

4 Partitioning Functions

In this section, we introduce the notion of *partitioning functions*. The notion abstracts out the information theoretic properties that are useful in the security proofs based on the partitioning techniques. Then, we proceed to recap the specific partitioning function that was given by [Jag15]. Then, we propose two new constructions of partitioning functions. The first one is obtained by introducing a simple but novel twist to the construction by [Jag15]. The second one is based on the affine-functions on random modulus. In the later sections, we will construct new lattice IBEs and VRFs based on these partitioning functions.

4.1 Definition

In the security proofs based on the partitioning technique [BB04b, Wat05], the simulations are successful only with noticeable probabilities. As observed by Waters [Wat05], this causes a subtle problem when considering the reduction to the decisional assumptions (such as the L-DDH). He resolved the problem by introducing the artificial abort step, where the simulator intentionally aborts with certain probability even when the simulation is successful. Later, Bellare and Ristenpart [BR09] showed that by requiring reasonable *upper bound* on the probability that the simulation is successful in addition to the *lower bound*, this step can be removed. In the subsequent work, Jager [Jag15] incorporated the idea of [BR09] into the notion of the admissible hash function [BB04b, CHKP10, FHPS13] to define *balanced admissible hash function*. The notion is a useful tool to perform the security proofs based on the partitioning technique. In addition, it is compatible with the decisional assumptions in the sense that it does not require the artificial abort step. Here, we define the notion of the partitioning function by slightly generalizing the balanced admissible hash function [Jag15].

Definition 7. *Let $\mathsf{F} = \{\ \mathsf{F}_\lambda : \mathcal{K}_\lambda \times \mathcal{X}_\lambda \to \{0,1\}\ \}$ be an ensemble of function families. We say that F is a partitioning function, if there exists an efficient algorithm $\mathsf{PrtSmp}(1^\lambda, Q, \epsilon)$, which takes as input polynomially bounded $Q = Q(\lambda) \in \mathbb{N}$ and noticeable $\epsilon = \epsilon(\lambda) \in (0, 1/2]$ and outputs K such that:*

1. *There exists $\lambda_0 \in \mathbb{N}$ such that*

$$\Pr\left[K \in \mathcal{K}_\lambda \ : \ K \xleftarrow{\$} \mathsf{PrtSmp}\left(1^\lambda, Q(\lambda), \epsilon(\lambda)\right) \ \right] = 1$$

   for all $\lambda > \lambda_0$. Here, λ_0 may depend on functions $Q(\lambda)$ and $\epsilon(\lambda)$.

2. *For $\lambda > \lambda_0$, there exists $\gamma_{\max}(\lambda)$ and $\gamma_{\min}(\lambda)$ that depend on $Q(\lambda)$ and $\epsilon(\lambda)$ such that for all $X^{(1)}, \ldots, X^{(Q)}, X^\star \in \mathcal{X}_\lambda$ with $X^\star \notin \{X^{(1)}, \ldots, X^{(Q)}\}$,*

$$\gamma_{\max}(\lambda) \geq \Pr\left[\ \mathsf{F}(K, X^{(1)}) = \cdots = \mathsf{F}(K, X^{(Q)}) = 1 \ \wedge \ \mathsf{F}(K, X^\star) = 0 \ \right] \geq \gamma_{\min}(\lambda) \tag{7}$$

   holds and the function $\tau(\lambda)$ defined as

$$\tau(\lambda) := \gamma_{\min}(\lambda) \cdot \epsilon(\lambda) - \frac{\gamma_{\max}(\lambda) - \gamma_{\min}(\lambda)}{2} \tag{8}$$

   is noticeable. The probability is taken over the choice of $K \xleftarrow{\$} \mathsf{PrtSmp}(1^\lambda, Q(\lambda), \epsilon(\lambda))$.

We call K the partitioning key and $\tau(\lambda)$ the quality of the partitioning function.

In the following, we often drop the subscript λ and denote F, $\mathcal{K}$, and $\mathcal{X}$ for the sake of simplicity. We remark that the term $\tau(\lambda)$ above, which may seem very specific, is inherited from [Jag15]. As explained in [Jag15], such a term appears typically in security analyses that follows the approach of Bellare and Ristenpart [BR09] (See also Lemma 6). Looking ahead, the quantity $\tau(\lambda)$ will directly affect the reduction cost of our IBEs and VRFs. The length of (the binary representation of) the partitioning key K will affect the efficiency of the resulting schemes. Therefore, we want the partitioning function F for the largest possible $\tau(\lambda)$ and the shortest possible partitioning key.

There are two main differences from the definition of [Jag15]. Firstly, we consider *any* function F, whereas they only considered a specific function (namely, $\mathsf{F}_{\mathsf{ADH}}$ in Sec. 4.2). Secondly, we explicitly add the condition regarding the domain correctness of the output of PrtSmp (the first condition), which was implicit in [Jag15].

Comparison with Programmable Hash Functions. Our notion of the partitioning function is also similar to the programmable hash function [HK08, ZCZ16]. The main difference is that whereas the notion of the programmable hash function is defined on specific algebraic structures such as (bilinear) groups [HK08] and lattices [ZCZ16], our definition is irrelevant to them. Since the security proofs of our IBEs and VRFs have the same information theoretic structure in common, we choose to decouple them from the underlying algebraic structures.

4.2 Construction from Admissible Hash Function

Here, we recap the result of Jager [Jag15] who constructed a specific partitioning function that he calls balanced admissible hash function. The result will be used in the next subsection to construct our first partitioning function. Let $k(\lambda) = \Theta(\lambda)$ and $\ell(\lambda) = \Theta(\lambda)$ be integers and let $\{C_k : \{0,1\}^k \to \{0,1\}^\ell\}_{k \in \mathbb{N}}$ be a family of error correcting codes with minimal distance ℓc for a constant $c \in (0, 1/2)$. Explicit constructions of such codes are given in [SS96, Zém01, Gol08] for instance. Let us define

$$\mathcal{K}_{\mathsf{ADH}} = \{0, 1, \bot\}^\ell \qquad \text{and} \qquad \mathcal{X}_{\mathsf{ADH}} = \{0, 1\}^k.$$

We define $\mathsf{F}_{\mathsf{ADH}}$ as

$$\mathsf{F}_{\mathsf{ADH}}(K, X) = \begin{cases} 0, & \text{if} \quad \forall i \in [\ell] : \quad C(X)_i = K_i \quad \vee \quad K_i = \perp \\ 1, & \text{otherwise} \end{cases}$$

where $C(X)_i$ and K_i are the i-th significant bit of $C(X)$ and K, respectively. Jager [Jag15] showed the following theorem.

Theorem 1. (Adapted from Theorem 1 in [Jag15].) There exists an efficient algorithm $\mathsf{AdmSmp}(1^\lambda, Q, \epsilon)$, which takes as input $Q \in \mathbb{N}$ and $\epsilon \in (0, 1/2]$ and outputs K with exactly η' components not equal to $\perp$, where

$$\eta' := \left\lfloor \frac{\log(2Q + Q/\epsilon)}{-\log(1 - c)} \right\rfloor,$$

such that Eq.(7) and (8) hold with respect to $\mathsf{F} := \mathsf{F}_{\mathsf{ADH}}$, $\mathsf{PrtSmp} := \mathsf{AdmSmp}$, and $\tau(\lambda) = 2^{-\eta'-1} \cdot \epsilon$. In particular, $\mathsf{F}_{\mathsf{ADH}}$ is a partitioning function.

4.3 Our Construction Based on Modified Admissible Hash Function

Here, we propose our first construction of the partitioning function $\mathsf{F}_{\mathsf{MAH}}$, which is obtained by modifying $\mathsf{F}_{\mathsf{ADH}}$ in the previous subsection. The advantage of $\mathsf{F}_{\mathsf{MAH}}$ is that it achieves much shorter partitioning keys compared with $\mathsf{F}_{\mathsf{ADH}}$. In particular, the length is $\omega(\log^2 \lambda)$ in $\mathsf{F}_{\mathsf{MAH}}$, whereas $\Theta(\lambda)$ in $\mathsf{F}_{\mathsf{ADH}}$. We will use the same notation as in Sec. 4.2. Let us introduce an integer $\eta(\lambda) = \omega(\log \lambda)$. $\eta(\lambda)$ can be set arbitrarily as long as it grows faster than $\log \lambda$. (See footnote in Sec. 1.) For our construction, we set

$$\mathcal{K}_{\mathsf{MAH}} = \{ \quad \mathsf{T} \subseteq [2\ell] \quad | \quad |\mathsf{T}| < \eta \quad \} \qquad \text{and} \qquad \mathcal{X}_{\mathsf{MAH}} = \{0,1\}^k.$$

We define $\mathsf{F}_{\mathsf{MAH}}$ as

$$\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = \begin{cases} 0, & \text{if } \mathsf{T} \subseteq \mathsf{S}(X) \\ 1, & \text{otherwise} \end{cases} \qquad \text{where} \qquad \mathsf{S}(X) = \{ \ 2i - C(X)_i \mid i \in [\ell] \ \}.$$

In the above, $C(X)_i$ is the i-th bit of $C(X) \in \{0,1\}^\ell$. See Fig. 1 in Sec. 2.1 for an illustrative example of S.

Lemma 7. *The function $\mathsf{F}_{\mathsf{MAH}}$ defined above is a partitioning function.*

Proof. To prove the lemma, we define $\mathsf{PrtSmp}_{\mathsf{MAH}}$ as follows. It uses the algorithm AdmSmp from the previous subsection as a subroutine.

$\mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q, \epsilon)$: It runs $\mathsf{AdmSmp}(1^\lambda, Q, \epsilon) \to K$ and sets

$$\mathsf{T} = \{ \ 2i - K_i \mid i \in [\ell], \ K_i \neq \perp \ \} \subseteq [2\ell],$$

where K_i is the i-th bit of K. It finally outputs T.

See Fig. 1 in Sec. 2.1 for an illustrative example of T. We first show that $\mathsf{PrtSmp}_{\mathsf{MAH}}$ satisfies the first property of Definition 7. By Theorem 1, $|\mathsf{T}| = \eta' = \lceil \log(2Q + Q/\epsilon) / \log(1 - c) \rceil$. To show

$\mathsf{T} \in \mathcal{K}_{\mathsf{MAH}}$ for all sufficiently large λ, it suffices to show $\eta'(\lambda) < \eta(\lambda)$ for all sufficiently large λ. This follows since

$$\eta'(\lambda) = \left\lfloor \frac{\log(2Q + Q/\epsilon)}{-\log(1-c)} \right\rfloor = O\left(\log(\mathsf{poly}(\lambda))\right) = O(\log \lambda) \qquad \text{and} \qquad \eta(\lambda) = \omega(\log \lambda)$$

when $Q(\lambda)$ is polynomially bounded and ϵ is noticeable for constant c. We next prove the second property. This follows from Theorem 1 and by the following observation:

$$\begin{aligned} \mathsf{F}_{\mathsf{ADH}}(K, X) = 0 \qquad & \Leftrightarrow \quad C(X)_i = K_i \quad \forall i \in [\ell] \text{ such that } K_i \neq \perp \\ & \Leftrightarrow \quad \mathsf{T} \subseteq \mathsf{S}(X) \\ & \Leftrightarrow \quad \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = 0. \end{aligned}$$

This completes the proof of Lemma 7. □

4.4 Our Construction Based on Affine Functions

Here, we propose our second construction of the partitioning function $\mathsf{F}_{\mathsf{AFF}}$. Compared to $\mathsf{F}_{\mathsf{MAH}}$, the function achieves an even shorter length of $\omega(\log \lambda)$ for the partitioning keys. Let $k(\lambda) = \Theta(\lambda)$ and $\eta(\lambda) = \omega(\log \lambda)$ be integers. For our construction, we set

$$\mathcal{K}_{\mathsf{AFF}} = \{0,1\}^{3\eta}, \qquad \mathcal{X}_{\mathsf{AFF}} = \{0,1\}^{k}$$

$\mathsf{F}_{\mathsf{AFF}}(K, X)$ is defined as

$$\mathsf{F}_{\mathsf{AFF}}(\ K = (\alpha, \beta, \rho),\ \ X\ ) = \begin{cases} 0, & \text{if} \quad \rho \neq 0 \quad \wedge \quad \alpha X + \beta \equiv 0 \mod \rho \\ 1, & \text{otherwise} \end{cases},$$

where $\alpha, \beta, \rho \in \{0,1\}^{\eta}$. Here, we slightly abuse the notation and identify a bit-string in $\{0,1\}^{\eta}$ with an integer in $[0, 2^{\eta} - 1]$ by its binary representation. Similarly, a bit-string in $\{0,1\}^{k}$ is identified with an integer in $[0, 2^{k} - 1]$.

Theorem 2. $\mathsf{F}_{\mathsf{AFF}}$ *defined above is a partitioning function.*

Proof. To prove the theorem, we define $\mathsf{PrtSmp}_{\mathsf{AFF}}$ as follows.

$\mathsf{PrtSmp}_{\mathsf{AFF}}(1^{\lambda}, Q, \epsilon)$: Let I be

$$I = \left\{ \text{ Prime } x \ \middle|\ \left\lceil \frac{k^2 Q}{\epsilon} \right\rceil + 1 \leq x \leq 4 \cdot \left\lceil \frac{k^2 Q}{\epsilon} \right\rceil \ \right\}.$$

It picks $\rho \xleftarrow{\$} I$ and $\alpha, \beta \xleftarrow{\$} [0, \rho - 1]$ and outputs $K = (\alpha, \beta, \rho)$.

We have to show that $\mathsf{PrtSmp}_{\mathsf{AFF}}$ has the two properties defined in Definition 7. We start with the first property. When $\epsilon(\lambda)$ is noticeable and $Q(\lambda)$ is polynomially bounded, we have

$$\rho \leq \frac{4k(\lambda)^2 Q(\lambda)}{\epsilon(\lambda)} + 1 = \mathsf{poly}(\lambda) \leq 2^{\eta(\lambda)} - 1$$

for all sufficiently large λ, where the latter inequality follows from $\eta(\lambda) = \omega(\log \lambda)$. Since $0 \leq \alpha, \beta \leq \rho - 1$, it follows that $K = (\alpha, \beta, \rho) \in ([0, 2^{\eta} - 1])^3 = \mathcal{K}_{\mathsf{AFF}}$ for all sufficiently large λ.

Next, we proceed to show the second property. Let us denote $\gamma(X^{(1)}, \ldots, X^{(Q)}, X^\star) := \Pr_K[\mathsf{F}_{\mathsf{AFF}}(K, X^{(1)}) = \cdots = \mathsf{F}_{\mathsf{AFF}}(K, X^{(Q)}) = 1 \ \wedge \ \mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0]$. We first obtain an upper bound for $\gamma(X^{(1)}, \ldots, X^{(Q)}, X^\star)$:

$$\begin{aligned}
\gamma(X^{(1)}, \ldots, X^{(Q)}, X^\star) &\leq \Pr_K[\mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0] \\
&= \Pr_{\alpha,\beta,\rho}\left[\alpha X^\star + \beta \equiv 0 \mod \rho\right] \\
&= \sum_{\bar\rho \in I} \Pr_\rho\left[\rho = \bar\rho\right] \cdot \Pr_{\alpha,\beta \xleftarrow{\$} [0, \bar\rho - 1]}\left[\alpha X^\star + \beta \equiv 0 \mod \bar\rho\right] \\
&= \underbrace{\sum_{\bar\rho \in I} \Pr_\rho\left[\rho = \bar\rho\right] \cdot \frac{1}{\bar\rho}}_{:=\gamma_{\max}}.
\end{aligned}$$

We have

$$\gamma_{\max} \geq \sum_{\bar\rho \in I} \Pr_\rho[\rho = \bar\rho] \cdot \frac{1}{4\lceil k^2 Q/\epsilon \rceil} = \frac{1}{4\lceil k^2 Q/\epsilon \rceil} \geq \frac{\epsilon}{5k^2 Q}$$

which we use later. Next, we show the lower bound. We have

$$\begin{aligned}
&\gamma(X^{(1)}, \ldots, X^{(Q)}, X^\star) \\
= \ &\Pr_K\left[\mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0\right] - \Pr_K\left[\ \mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0 \ \wedge \ \left(\vee_{j \in [Q]} \mathsf{F}_{\mathsf{AFF}}(K, X^{(j)}) = 0\right)\ \right] \\
= \ &\Pr_K\left[\mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0\right] - \Pr_K\left[\ \vee_{j \in [Q]} \left(\mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0 \ \wedge \ \mathsf{F}_{\mathsf{AFF}}(K, X^{(j)}) = 0\right)\ \right] \\
\geq \ &\underbrace{\Pr_K\left[\mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0\right]}_{=\gamma_{\max}} - \sum_{j \in [Q]} \underbrace{\Pr_K\left[\ \mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0 \ \wedge \ \mathsf{F}_{\mathsf{AFF}}(K, X^{(j)}) = 0\ \right]}_{:=\gamma_j}.
\end{aligned}$$

It suffices to show the upper bound on γ_j.

$$\begin{aligned}
\gamma_j &= \Pr_K\left[\ \mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0 \ \wedge \ \mathsf{F}_{\mathsf{AFF}}(K, X^{(j)}) = 0\ \right] \\
&= \Pr_{\alpha,\beta,\rho}\left[\ \alpha X^\star + \beta \equiv 0 \mod \rho \quad \wedge \quad \alpha X^{(j)} + \beta \equiv 0 \mod \rho\ \right] \\
&= \Pr_{\alpha,\beta,\rho}\left[\ \alpha X^\star + \beta \equiv 0 \mod \rho \quad \wedge \quad \alpha \cdot \left(X^\star - X^{(j)}\right) \equiv 0 \mod \rho\ \right] \\
&= \Pr_{\alpha,\beta,\rho}\left[\ \alpha X^\star + \beta \equiv 0 \mod \rho \quad \wedge \quad \left(\alpha \equiv 0 \mod \rho \ \vee \ X^\star - X^{(j)} \equiv 0 \mod \rho\right)\ \right] \\
&\leq \underbrace{\Pr_{\alpha,\beta,\rho}\left[\ \alpha X^\star + \beta \equiv 0 \mod \rho \quad \wedge \quad \alpha \equiv 0 \mod \rho\ \right]}_{:=\gamma'_j} \\
&\quad + \underbrace{\Pr_{\alpha,\beta,\rho}\left[\ \alpha X^\star + \beta \equiv 0 \mod \rho \ \wedge \ X^\star - X^{(j)} \equiv 0 \mod \rho\ \right]}_{:=\gamma''_j}
\end{aligned}$$

The last equality follows from the fact that ρ is prime. We then bound γ'_j and γ''_j. We have

$$\begin{aligned}
\gamma'_j &= \Pr_{\alpha,\beta,\rho}\left[\ \alpha X^\star + \beta \equiv 0 \mod \rho \quad \wedge \quad \alpha \equiv 0 \mod \rho\ \right] \\
&= \Pr_{\alpha,\beta,\rho}\left[\ \alpha \equiv 0 \mod \rho \quad \wedge \quad \beta \equiv 0 \mod \rho\ \right]
\end{aligned}$$

$$\begin{aligned}
&= \sum_{\bar{\rho} \in I} \Pr_{\rho}\left[\rho = \bar{\rho}\right] \cdot \frac{1}{\bar{\rho}^2} \\
&\leq \sum_{\bar{\rho} \in I} \Pr_{\rho}\left[\rho = \bar{\rho}\right] \cdot \left(\frac{\epsilon}{k^2 Q}\right)^2 \\
&= \frac{\epsilon^2}{k^4 Q^2} \\
&\leq \frac{\epsilon^2}{18 k^2 Q^2}
\end{aligned}$$

where the last inequality holds for $k \geq 5$. To bound γ''_j, we introduce a set J as

$$J := \left\{ \text{ Prime } x \;\middle|\; x \text{ divides } |X^\star - X^{(j)}| \right\}.$$

Since $|X^\star - X^{(j)}| \leq 2^k$, it can be seen that $|J| \leq \log 2^k = k$. This bound can be obtained by replacing all prime factors of $|X^\star - X^{(j)}|$ with 2. We have

$$\begin{aligned}
\gamma''_j &= \Pr_{\alpha,\beta,\rho}\left[\, \alpha X^\star + \beta \equiv 0 \mod \rho \;\wedge\; X^\star - X^{(j)} \equiv 0 \mod \rho \,\right] \\
&= \sum_{\bar{\rho} \in I} \Pr_{\rho \overset{\$}{\leftarrow} I}\left[\rho = \bar{\rho}\right] \cdot \Pr_{\alpha,\beta \overset{\$}{\leftarrow} [0,\bar{\rho}-1]}\left[\, \alpha X^\star + \beta \equiv 0 \mod \bar{\rho} \;\wedge\; X^\star - X^{(j)} \equiv 0 \mod \bar{\rho} \,\right] \\
&= \sum_{\bar{\rho} \in I \cap J} \Pr_{\rho \overset{\$}{\leftarrow} I}\left[\rho = \bar{\rho}\right] \cdot \Pr_{\alpha,\beta \overset{\$}{\leftarrow} [0,\bar{\rho}-1]}\left[\, \alpha X^\star + \beta \equiv 0 \mod \bar{\rho} \,\right] \qquad (9) \\
&= \sum_{\bar{\rho} \in I \cap J} \left(\frac{1}{|I|}\right) \cdot \left(\frac{1}{\bar{\rho}}\right) \\
&\leq \frac{|I \cap J|}{|I|} \cdot \frac{\epsilon}{k^2 Q} \\
&\leq k \cdot \left(\frac{\log(\lceil k^2 Q/\epsilon \rceil)}{\lceil k^2 Q/\epsilon \rceil}\right) \cdot \frac{\epsilon}{k^2 Q} \qquad (10) \\
&\leq k \cdot \left(\frac{\log(1 + k^2 Q/\epsilon)}{k^2 Q/\epsilon}\right) \cdot \frac{\epsilon}{k^2 Q} \\
&= \frac{\epsilon^2 \cdot \log(1 + k^2 Q/\epsilon)}{k^3 Q^2} \\
&\leq \frac{\epsilon^2}{18 k^2 Q^2} \qquad (11)
\end{aligned}$$

where Eq.(9) follows since $X^\star - X^{(j)} \equiv 0 \mod \bar{\rho}$ if and only if $\bar{\rho} \in I \cap J$, Eq.(10) follows from Lemma 5, and Eq.(11) follows from $18 \log(1 + k^2 Q/\epsilon) \leq k$, which holds for sufficiently large λ since

$$\log\left(1 + \frac{k(\lambda)^2 Q(\lambda)}{\epsilon(\lambda)}\right) = \log(\mathsf{poly}(\lambda)) = O(\log \lambda), \quad \text{and} \quad k(\lambda) = \Theta(\lambda).$$

Putting the pieces together, we have

$$\gamma(X^{(1)}, \ldots, X^{(Q)}, X^\star) \;\; \geq \;\; \gamma_{\max} - \sum_{j \in Q} \left(\frac{\epsilon^2}{18 k^2 Q^2} + \frac{\epsilon^2}{18 k^2 Q^2}\right)$$

$$\geq \underbrace{\gamma_{\max} - \frac{\epsilon^2}{9k^2Q}}_{:=\gamma_{\min}}.$$

It remains to show that $\gamma_{\min}\epsilon - (\gamma_{\max} - \gamma_{\min})/2$ is noticeable. We have

$$\begin{aligned}\gamma_{\min}\epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2} &= \gamma_{\max}\epsilon - (2\epsilon + 1)\left(\frac{\epsilon^2}{18k^2Q}\right)\\ &\geq \frac{\epsilon^2}{5k^2Q} - 3 \cdot \left(\frac{\epsilon^2}{18k^2Q}\right)\\ &\geq \frac{\epsilon^2}{30k^2Q}.\end{aligned}$$

The quantity is noticeable and this completes the proof of Theorem 2. □

5 Our IBE Schemes

In this section, we give a generic construction of an adaptively secure lattice based IBE from a partitioning function. Our generic construction requires the underlying partitioning function to be compatible (in some sense) with the structure of lattices. In the following, we first formalize the requirement by giving the definition of compatibility. Then, we show that $\mathsf{F}_{\mathsf{MAH}}$ and $\mathsf{F}_{\mathsf{AFF}}$ are compatible in this sense. Finally, we show the generic construction of IBE and prove its security.

5.1 Compatible Algorithms for Partitioning Functions

The following definition gives a sufficient condition for partitioning functions to be useful for constructing adaptively secure IBE schemes.

Definition 8. *We say that the* deterministic *algorithms* ($\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval}$) *are δ-compatible with a function family* $\{\mathsf{F} : \mathcal{K} \times \mathcal{X} \to \{0,1\}\}$ *if they are efficient and satisfy the following properties:*

- $\mathsf{Encode}(\ K \in \mathcal{K}\ ) \to \kappa \in \{0,1\}^u$
- $\mathsf{PubEval}\left(\ X \in \mathcal{X},\ \{\mathbf{B}_i \in \mathbb{Z}_q^{n\times m}\}_{i\in[u]}\ \right) \to \mathbf{B}_X \in \mathbb{Z}_q^{n\times m}$
- $\mathsf{TrapEval}\left(\ K \in \mathcal{K},\ X \in \mathcal{X},\ \mathbf{A} \in \mathbb{Z}_q^{n\times m},\ \{\mathbf{R}_i \in \mathbb{Z}^{m\times m}\}_{i\in[u]}\ \right) \to \mathbf{R}_X \in \mathbb{Z}^{m\times m}$
  We require that the following holds:

  $$\mathsf{PubEval}\left(X, \{\mathbf{A}\mathbf{R}_i + \kappa_i\mathbf{G}\}_{i\in[u]}\right) = \mathbf{A}\mathbf{R}_X + \mathsf{F}(K,X)\cdot\mathbf{G}$$

  where $\kappa_i \in \{0,1\}$ is the i-th bit of $\kappa = \mathsf{Encode}(K) \in \{0,1\}^u$. Furthermore, if $\mathbf{R}_i \in \{-1,0,1\}^{m\times m}$ for all $i \in [u]$, we have $\|\mathbf{R}_X\|_\infty \leq \delta$.

It is possible to obtain compatible algorithms for any partitioning functions, including ours, by directly using the fully key homomorphic algorithm in [BGG$^+$14]. However, if we apply the algorithm naively, it will end up with super-polynomial δ, which is undesirable. In Sec. 5.2 and 5.3, we will provide δ-compatible algorithms for $\mathsf{F}_{\mathsf{MAH}}$ and $\mathsf{F}_{\mathsf{AFF}}$ for polynomial δ by carefully applying the idea from [GV15]. In Sec. 5.4, we obtain new lattice IBE schemes using these algorithms.

5.2 Compatible Algorithms for $\mathsf{F}_{\mathsf{MAH}}$

Here, we show compatible algorithms for $\mathsf{F}_{\mathsf{MAH}}$. Before describing the algorithms, we state and prove the following lemma, which will be used in the following. The lemma is a variant of Lemma 6 in [Yam16] and can be proven based on the idea from [GV15]. The proof appears in Appendix A.

Lemma 8 (Homomorphic Multiplication)**.** *Let $d = d(\lambda)$ be a natural number. There exist two efficient deterministic algorithms $\mathsf{PubMult}_d$ and $\mathsf{TrapMult}_d$ with the following properties:*

- $\mathsf{PubMult}_d\left(\{\mathbf{B}_i\}_{i\in[d]}\right) \to \mathbf{B}_d^{\times} \in \mathbb{Z}_q^{n\times m}$. *Here,* $\mathbf{B}_i \in \mathbb{Z}_q^{n\times m}$.
- $\mathsf{TrapMult}_d\left(\mathbf{A}, \{\mathbf{R}_i, x_i\}_{i\in[d]}\right) \to \mathbf{R}_d^{\times} \in \mathbb{Z}^{m\times m}$. *Here,* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\|\mathbf{R}_i\|_\infty \le \delta$, $x_i \in \{0,1\}$. *Furthermore, we have*

$$\mathsf{PubMult}_d\left(\{\mathbf{A}\mathbf{R}_i + x_i\mathbf{G}\}_{i\in[d]}\right) \quad = \quad \mathbf{A}\mathbf{R}_d^{\times} + x_1\cdots x_d\cdot\mathbf{G} \tag{12}$$

and $\|\mathbf{R}_d^{\times}\|_\infty \le md\delta$.

Compatible Algorithms for $\mathsf{F}_{\mathsf{MAH}}$. Here, we provide compatible algorithms $(\mathsf{Encode}_{\mathsf{MAH}}, \mathsf{PubEval}_{\mathsf{MAH}}, \mathsf{TrapEval}_{\mathsf{MAH}})$ for $\mathsf{F}_{\mathsf{MAH}}$. Let k, ℓ, η, and S be as in Sec. 4.3. We also introduce $\zeta(\lambda) := \lceil\log(2\ell+1)\rceil$ and set $u(\lambda) := \eta(\lambda)\zeta(\lambda)$. Note that by our choice of ζ, any integer in $[0, 2\ell]$ can be represented as a string in $\{0,1\}^\zeta$ by its binary representation. In the following, for notational convenience, we use the index space $[\eta]\times[\zeta]$ instead of $[u]$. For instance, the input to $\mathsf{PubEval}$ is denoted as X and $\{\mathbf{B}_{i,j}\}_{(i,j)\in[\eta]\times[\zeta]}$, instead of X and $\{\mathbf{B}\}_{i\in[u]}$.

$\mathsf{Encode}_{\mathsf{MAH}}(\mathsf{T})$: Given $\mathsf{T} \subseteq [2\ell]$, it first parses it as $\mathsf{T} \to \{t_1, \ldots, t_{\eta'}\}$ where $\eta' < \eta$ and $t_i \in [2\ell]$, and it sets $t_{\eta'+1} = \cdots = t_\eta = 0$. It then concatenates the binary representations of $t_1, \ldots, t_\eta$ to obtain a bit string κ in $\{0,1\}^{\eta\zeta} = \{0,1\}^u$. Finally, it outputs κ.

$\mathsf{PubEval}_{\mathsf{MAH}}\left(X, \{\mathbf{B}_{i,j}\}_{(i,j)\in[\eta]\times[\zeta]}\right)$: It first computes $\mathsf{S}(X) = \{s_1, \ldots, s_\ell\} \subset [2\ell]$. It also sets $s_{\ell+1} = 0$. Let $s_{i,j} \in \{0,1\}$ be the j-th bit of the binary representation of s_i. It then proceeds as follows:

1. For $(i,j,i') \in [\eta]\times[\zeta]\times[\ell+1]$, it sets $\mathbf{V}_{i,j,i'} = \begin{cases} \mathbf{G} - \mathbf{B}_{i,j} & \text{if } s_{i',j} = 0 \\ \mathbf{B}_{i,j} & \text{if } s_{i',j} = 1. \end{cases}$
2. For $(i,i') \in [\eta]\times[\ell+1]$, it computes $\mathbf{V}_{i,i'} := \mathsf{PubMult}_\zeta(\{\mathbf{V}_{i,j,i'}\}_{j\in[\zeta]})$.
3. For $i \in [\eta]$, it computes $\mathbf{V}_i := \mathbf{G} - \mathsf{PubMult}_{\ell+1}(\{\mathbf{G} - \mathbf{V}_{i,i'}\}_{i'\in[\ell+1]})$.
4. Finally, it computes the output $\mathbf{B}_X$ as $\mathbf{B}_X = \mathbf{G} - \mathsf{PubMult}_\eta\left(\{\mathbf{V}_i\}_{i\in[\eta]}\right)$.

$\mathsf{TrapEval}_{\mathsf{MAH}}\left(\mathsf{T}, X, \mathbf{A}, \{\mathbf{R}_{i,j}\}_{(i,j)\in[\eta]\times[\zeta]}\right)$: It first computes $\mathsf{S}(X) = \{s_1, \ldots, s_\ell\} \subset [2\ell]$ and parses $\mathsf{T} \to (t_1, \ldots, t_{\eta'}) \subseteq [2\ell]$, where $\eta' < \eta$ and $t_i \in [2\ell]$. It then sets $s_{\ell+1} := 0$ and $t_{\eta'+1} = \cdots = t_\eta = 0$. In the following, let $s_{i,j}$ and $t_{i,j}$ be the j-th bit of the binary representation of s_i and t_i, respectively. It then proceeds as follows:

1. For $(i,j,i') \in [\eta]\times[\zeta]\times[\ell+1]$, it sets $\begin{cases} \mathbf{S}_{i,j,i'} := -\mathbf{R}_{i,j},\ b_{i,j,i'} := 1 - t_{i,j} & \text{if } s_{i',j} = 0 \\ \mathbf{S}_{i,j,i'} := \mathbf{R}_{i,j},\quad b_{i,j,i'} := t_{i,j} & \text{if } s_{i',j} = 1. \end{cases}$
2. For $(i,i') \in [\eta]\times[\ell+1]$, it computes $b_{i,i'} := \prod_{j\in[\zeta]} b_{i,j,i'}$ and $\mathbf{S}_{i,i'} := \mathsf{TrapMult}_\zeta(\mathbf{A}, \{\mathbf{S}_{i,j,i'}, b_{i,j,i'}\}_{j\in[\zeta]})$.

3. For $i \in [\eta]$, it computes $\mathbf{S}_i = -\mathsf{TrapMult}_{\ell+1}(\mathbf{A}, \{-\mathbf{S}_{i,i'}, 1 - b_{i,i'}\}_{i' \in [\ell+1]})$ and $b_i = 1 - \prod_{i' \in [\ell+1]}(1 - b_{i,i'})$.
4. Finally, it computes $\mathbf{R}_X = -\mathsf{TrapMult}_\eta \left(\mathbf{A}, \{\mathbf{S}_i, b_i\}_{i \in [\eta]}\right)$ and $b_X = 1 - \prod_{i \in [\eta]} b_i$ and outputs $\mathbf{R}_X$.

Lemma 9. $(\mathsf{Encode}_{\mathsf{MAH}}, \mathsf{PubEval}_{\mathsf{MAH}}, \mathsf{TrapEval}_{\mathsf{MAH}})$ *defined above are* $m^3u(\ell+1)$*-compatible algorithms for* $\mathsf{F}_{\mathsf{MAH}}$.

Proof. We have to show $\|\mathbf{R}_X\|_\infty \leq m^3u(\ell+1)$ and

$$\mathsf{PubEval}\left(X, \{\mathbf{A}\mathbf{R}_{i,j} + t_{i,j}\mathbf{G}\}_{(i,j) \in [\eta] \times [\zeta]}\right) = \mathbf{A}\mathbf{R}_X + \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) \cdot \mathbf{G}$$

for $\mathsf{TrapEval}\left(\mathsf{T}, X, \mathbf{A}, \{\mathbf{R}_{i,j}\}_{(i,j) \in [\eta] \times [\zeta]}\right) \to \mathbf{R}_X$ when $\mathbf{R}_{i,j} \in \{-1, 0, 1\}^{m \times m}$. We start with the former. By repeatedly applying Lemma 8, it can be seen that when $\mathbf{B}_{i,j} = \mathbf{A}\mathbf{R}_{i,j} + t_{i,j}\mathbf{G}$, we have

$$\begin{array}{cccc} \mathbf{V}_{i,j,i'} = \mathbf{A}\mathbf{S}_{i,j,i'} + b_{i,j,i'}\mathbf{G}, & \mathbf{V}_{i,i'} = \mathbf{A}\mathbf{S}_{i,i'} + b_{i,i'}\mathbf{G}, & \mathbf{V}_i = \mathbf{A}\mathbf{S}_i + b_i\mathbf{G}, & \mathbf{B}_X = \mathbf{A}\mathbf{R}_X + b_X\mathbf{G}, \\ b_{i,j,i'} \in \{0,1\}, & b_{i,i'} \in \{0,1\}, & b_i \in \{0,1\}, & b_X \in \{0,1\}, \\ \|\mathbf{S}_{i,j,i'}\|_\infty \leq 1, & \|\mathbf{S}_{i,i'}\|_\infty \leq m\zeta, & \|\mathbf{S}_i\|_\infty \leq m^2\zeta(\ell+1), & \|\mathbf{R}_X\|_\infty \leq m^3\eta\zeta(\ell+1), \end{array}$$

as desired. To prove the latter, it suffices to show $b_X = \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X)$. We have

$$\begin{aligned} b_X &= 1 - \prod_{i \in [\eta]} b_i \\ &= 1 - \prod_{i \in [\eta]} \left(1 - \prod_{i' \in [\ell+1]} (1 - b_{i,i'})\right) \\ &= 1 - \prod_{i \in [\eta]} \left(1 - \prod_{i' \in [\ell+1]} \left(1 - \prod_{j \in [\zeta]} b_{i,j,i'}\right)\right) \\ &= \neg \bigwedge_{i \in [\eta]} \left(\neg \bigwedge_{i' \in [\ell+1]} \left(\neg \bigwedge_{j \in [\zeta]} b_{i,j,i'}\right)\right) && (13) \\ &= \neg \left(\bigwedge_{i \in [\eta]} \bigvee_{i' \in [\ell+1]} \bigwedge_{j \in [\zeta]} b_{i,j,i'}\right) \\ &= \neg \left(\bigwedge_{i \in [\eta]} \bigvee_{i' \in [\ell+1]} \bigwedge_{j \in [\zeta]} \left(t_{i,j} \overset{?}{=} s_{i',j}\right)\right) && (14) \\ &= \neg \left(\bigwedge_{i \in [\eta]} \bigvee_{i' \in [\ell+1]} \left(t_i \overset{?}{=} s_{i'}\right)\right) \\ &= \neg \left(\bigwedge_{i \in [\eta]} \left(t_i \overset{?}{\in} \mathsf{S}(X) \cup \{0\}\right)\right) && (15) \\ &= \neg \left(\bigwedge_{i \in [\eta']} \left(t_i \overset{?}{\in} \mathsf{S}(X)\right)\right) && (16) \end{aligned}$$

$$
\begin{aligned}
&= \neg\left(\mathsf{T} \overset{?}{\subseteq} \mathsf{S}(X)\right) \\
&= \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X)
\end{aligned}
$$

where we regard $b_{i,j,i'}$ as boolean values rather than integers in Eq.(13) since they are in $\{0,1\}$, Eq.(14) follows from $b_{i,j,i'} = 1$ if and only if $t_{i,j} = s_{i',j}$, Eq.(15) follows from $\mathsf{S}(X) = \{s_1, \ldots, s_\ell\}$ and $s_{\ell+1} = 0$, and Eq.(16) follows from $t_i \neq 0$ for $i \leq \eta'$ and $t_i = 0$ for $i \geq \eta' + 1$. This completes the proof of Lemma 9. □

5.3 Compatible Algorithms for $\mathsf{F}_{\mathsf{AFF}}$

Here, we provide compatible algorithms for $\mathsf{F}_{\mathsf{AFF}}$ with $\delta = \mathsf{poly}(n)$ here. Let k and η be as in Sec. 4.4. We also define $u := 3\eta$. To start with, we state the following lemma, which shows that $\mathsf{F}_{\mathsf{AFF}}$ can be computed in $\mathbf{NC}^1$. The proof is fairly standard given the result of Beam, Cook, and Hoover [BCH86], who showed integer division is in $\mathbf{NC}^1$.

Lemma 10. *The function $\mathsf{F}_{\mathsf{AFF}} : \{0,1\}^{u+k} \to \{0,1\}$ can be computed by a circuit with depth $O(\log(u+k)) = O(\log n)$. Furthermore, the description of the circuit can be computed in time $\mathsf{poly}(u,k)$.*

Proof. Let us parse the input as $(K = (\alpha, \beta, \rho) \in \{0,1\}^{3\eta}, X \in \{0,1\}^k)$. From the input, $\alpha X + \beta$ can be computed by $O(\log(u+k))$-depth circuits, which follows from the well-known fact that the addition, subtraction, and multiplication over the integers can be computed in $\mathbf{NC}^1$. In the next step, we apply the result of Beam, Cook, and Hoover [BCH86], who showed that $\lfloor a/b \rfloor \in \mathbb{Z}$ can be computed in $\mathbf{NC}^1$ given the binary representations of two natural numbers a and b. This implies that $(a \mod b)$ can also be computed in $\mathbf{NC}^1$, since $(a \mod b) = a - b \cdot \lfloor a/b \rfloor$. Using the result, one can compute $(\alpha X + \beta \mod \rho)$ by an additional $O(\log(u+k))$-depth circuit because $\alpha X + \beta \leq 2^{\eta+k+1} \in \{0,1\}^{\eta+k+2}$ and $\rho \in \{0,1\}^\eta$. (In the case of $\rho = 0$, it outputs 1.) Finally, it outputs 0 if $(\alpha X + \beta \mod \rho) = 0$ and otherwise 1. The final step can also be implemented by $O(\log(u+k))$-depth circuits. Summing up the above discussions, it follows that $\mathsf{F}_{\mathsf{AFF}}$ can be computed by $O(\log(u+k))$-depth circuits. This completes the proof of Lemma 10. □

We further state the following lemma, which is needed to obtain $\mathsf{poly}(n)$-compatible algorithm for $\mathsf{F}_{\mathsf{AFF}}$. The lemma can be obtained by the combination of the techniques in [GV15] and Barrington's Theorem [Bar89].

Lemma 11. *Let $\ell := \ell(n) \in \mathbb{N}$ be a natural number and $\mathsf{F} : \{0,1\}^\ell \to \{0,1\}$ be a function that can be computed by a Boolean NAND circuit with depth d. There exist two deterministic algorithms $\mathsf{NC^1PubEval}$ and $\mathsf{NC^1TrapEval}$ with the following properties.*

- $\mathsf{NC^1PubEval}\left(\mathsf{F}, \{\mathbf{B}_i\}_{i\in[\ell]}\right) \to \mathbf{B}_\mathsf{F} \in \mathbb{Z}_q^{n\times m}$. *Here,* $\mathbf{B}_i \in \mathbb{Z}_q^{n\times m}$.
- $\mathsf{NC^1TrapEval}\left(\mathsf{F}, \mathbf{A}, \{\mathbf{R}_i, x_i\}_{i\in[\ell]}\right) \to \mathbf{R}_\mathsf{F} \in \mathbb{Z}^{m\times m}$. *Here,* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{R}_i \in \{-1,0,1\}^{m\times m}$, *and* $x_i \in \{0,1\}$. *Furthermore, we have*

$$
\begin{aligned}
\mathsf{NC^1PubEval}\left(\mathsf{F}, \{\mathbf{A}\mathbf{R}_i + x_i\mathbf{G}\}_{i\in[\ell]}\right) &= \mathbf{A}\mathbf{R}_\mathsf{F} + \mathsf{F}(x)\cdot\mathbf{G} \\
\textit{and} \qquad \|\mathbf{R}_\mathsf{F}\|_\infty &\leq (3m+1)\cdot 4^d,
\end{aligned}
$$

where $x \in \{0,1\}^\ell$ is the concatenation of $x_1, \ldots, x_\ell$.

- *The running time of* $\mathsf{NC^1PubEval}$ *and* $\mathsf{NC^1TrapEval}$ *are* $\mathsf{poly}(\ell, 4^d, n, m, \log q)$*. In particular, if* $d = O(\log n)$*, it runs in polynomial time.*

The proof of lemma 11 will appear in Appendix B.

Compatible Algorithms for $\mathsf{F_{AFF}}$. We now use the algorithms in lemma 11 to construct compatible algorithms $(\mathsf{Encode_{AFF}}, \mathsf{PubEval_{AFF}}, \mathsf{TrapEval_{AFF}})$ for $\mathsf{F_{AFF}}$ as follows.

$\mathsf{Encode_{AFF}}(K)$: Given an input $K \in \{0,1\}^u$, it outputs the same bit-string $\kappa := K$.

$\mathsf{PubEval_{AFF}}(X, \{\mathbf{B}_i\}_{i\in[u]})$: It first sets $\mathbf{B}_{i+u} := X_i \cdot \mathbf{G}$ for $i \in [k]$, where $X_i \in \{0,1\}$ is the i-th bit of X. It then computes $\mathbf{B}_X := \mathsf{NC^1PubEval}(\mathsf{F_{AFF}}, \{\mathbf{B}_i\}_{i\in[u+k]})$ and returns the output.

$\mathsf{TrapEval_{AFF}}(K, X, \mathbf{A}, \{\mathbf{R}_i\}_{i\in[u]})$: It first sets $x_i := K_i$ for $i \in [u]$. It also sets $\mathbf{R}_{i+u} := \mathbf{0}_{n\times m}$ and $x_{i+u} = X_i$ for $i \in [k]$, where K_j and X_j are the j-th bit of K and X, respectively. It then computes $\mathbf{R}_X := \mathsf{NC^1TrapEval}(\mathsf{F_{AFF}}, \mathbf{A}, \{\mathbf{R}_i, x_i\}_{i\in[u+k]})$ and returns the output.

Lemma 12. $(\mathsf{Encode_{AFF}}, \mathsf{PubEval_{AFF}}, \mathsf{TrapEval_{AFF}})$ *defined above are δ-compatible algorithms for* $\mathsf{F_{AFF}}$ *for some fixed polynomial* $\delta = \mathsf{poly}(n)$*.*

Proof. We prove that these algorithms satisfy the desired properties. Let us assume that $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + K_i\mathbf{G}$ for $i \in [u]$. We further assume that $x = \{x_i\}_{i\in[u+k]}$, $\{\mathbf{B}_i\}_{i\in[u+1,u+k]}$, and $\{\mathbf{R}_i\}_{i\in[u+1,u+k]}$ are computed as specified. Then, we have $\mathsf{F_{AFF}}(x) = \mathsf{F_{AFF}}(K, X)$ and $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + x_i\mathbf{G}$ for $i \in [u+k]$. Therefore, the property of $\mathsf{NC^1PubEval}$ and $\mathsf{NC^1TrapEval}$ imply that $\mathsf{PubEval_{AFF}}(X, \{\mathbf{B}_i\}_{i\in[u]}) = \mathbf{A}\mathbf{R}_X + \mathsf{F_{AFF}}(x)\mathbf{G} = \mathbf{A}\mathbf{R}_X + \mathsf{F_{AFF}}(K, X)\mathbf{G}$. By Lemma 11 and the fact that $d = O(\log n)$, it also follows that $\|\mathbf{R}_X\|_\infty \le (3m+1)\cdot 4^d = \mathsf{poly}(n)$. This completes the proof of Lemma 12. □

5.4 Construction

Here, we construct an IBE scheme based on a partitioning function $\mathsf{F} : \mathcal{K} \times \mathcal{X} \to \{0,1\}$ with associating δ-compatible algorithms $(\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval})$. We assume $\mathcal{X} = \mathcal{ID} = \{0,1\}^k$, where $\mathcal{ID}$ is the identity space of the scheme. If a collision resistant hash $\mathsf{CRH} : \{0,1\}^* \to \{0,1\}^k$ is available, we can use any bit-string as an identity. For simplicity, we let the message space of the scheme be $\{0,1\}$. For the multi-bit variant, we refer to Sec. 5.7. Our scheme can be instantiated with $\mathsf{F_{MAH}}$ and $\mathsf{F_{AFF}}$, which would lead to schemes with efficiency and security trade-offs. We compare the resulting schemes with existing schemes in Sec. 7. (See also Table 1 in Sec. 1.)

$\mathsf{Setup}(1^\lambda)$: On input 1^λ, it sets the parameters n, m, q, σ, α, and α' as specified in Sec. 5.5, where q is a prime number. Then, it picks random matrices $\mathbf{B}_0, \mathbf{B}_i \stackrel{\$}{\leftarrow} \mathbb{Z}_q^{n\times m}$ for $i \in [u]$ and a vector $\mathbf{u} \stackrel{\$}{\leftarrow} \mathbb{Z}_q^n$. It also picks $(\mathbf{A}, \mathbf{A}_{\sigma_0}^{-1}) \stackrel{\$}{\leftarrow} \mathsf{TrapGen}(1^n, 1^m, q)$ such that $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\sigma_0 = \omega(\sqrt{n \log q \log m})$. It finally outputs

$$\mathsf{mpk} = \big(\ \mathbf{A},\ \mathbf{B}_0,\ \{\mathbf{B}_i\}_{i\in[u]},\ \mathbf{u}\ \big) \qquad \text{and} \qquad \mathsf{msk} = \mathbf{A}_{\sigma_0}^{-1}.$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$: Given an identity ID, it first computes

$$\mathsf{PubEval}\big(\ \mathsf{ID},\ \{\mathbf{B}_i\}_{i\in[u]}\ \big) \to \mathbf{B}_{\mathsf{ID}} \in \mathbb{Z}_q^{n\times m}.$$

It then computes $[\mathbf{A}\|\mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}]_\sigma^{-1}$ from $\mathbf{A}_{\sigma_0}^{-1}$ and samples

$$\mathbf{e} \stackrel{\$}{\leftarrow} [\mathbf{A}\|\mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}]_\sigma^{-1}(\mathbf{u}).$$

Then, it returns $\mathsf{sk_{ID}} = \mathbf{e} \in \mathbb{Z}^{2m}$. Note that we have $[\mathbf{A}\|\mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}] \cdot \mathbf{e} = \mathbf{u} \mod q$.

$\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0,1\}$, it first computes $\mathsf{PubEval}(\mathsf{ID}, \{\mathbf{B}_i\}_{i\in[u]}) \to \mathbf{B}_{\mathsf{ID}}$. It then picks $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $x_0 \xleftarrow{\$} D_{\mathbb{Z},\alpha q}$, $\mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha' q}$ and computes

$$c_0 = \mathbf{s}^\top \mathbf{u} + x_0 + \mathsf{M} \cdot \lceil q/2 \rceil \in \mathbb{Z}_q, \qquad \mathbf{c}_1^\top = \mathbf{s}^\top [\mathbf{A} \| \mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}] + [\mathbf{x}_1^\top \| \mathbf{x}_2^\top] \in \mathbb{Z}_q^{2m}.$$

Finally, it returns the ciphertext $\mathsf{ct} = (c_0, \mathbf{c}_1)$.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{ct})$: To decrypt a ciphertext $\mathsf{ct} = (c_0, \mathbf{c}_1)$ using a private key $\mathsf{sk}_{\mathsf{ID}} := \mathbf{e}$, it first computes

$$w = c_0 - \mathbf{c}_1^\top \cdot \mathbf{e} \in \mathbb{Z}_q.$$

Then it returns 1 if $|w - \lceil q/2 \rceil| < \lceil q/4 \rceil$ and 0 otherwise.

5.5 Correctness and Parameter Selection

Here, we prove the correctness and fix the parameters for the scheme. When the cryptosystem is operated as specified, we have during decryption,

$$w = c_0 - \mathbf{c}_1^\top \cdot \mathbf{e} = \mathsf{M} \cdot \lceil q/2 \rceil + \underbrace{x_0 - [\mathbf{x}_1^\top \| \mathbf{x}_2^\top] \cdot \mathbf{e}}_{\text{error term}}.$$

Lemma 13. *Assuming $\alpha' > \alpha$, the error term is bounded by $O(\alpha' \sigma m q)$ with overwhelming probability.*

Proof. With overwhelming probability, we have

$$\begin{aligned} |x_0 - [\mathbf{x}_1^\top \| \mathbf{x}_2^\top] \cdot \mathbf{e}| &\le |x_0| + |[\mathbf{x}_1^\top \| \mathbf{x}_2^\top] \cdot \mathbf{e}| \\ &\le |x_0| + \|[\mathbf{x}_1^\top \| \mathbf{x}_2^\top]\| \cdot \|\mathbf{e}\| \\ &\le \alpha q \sqrt{m} + (\alpha' q \sqrt{2m}) \cdot (\sigma \sqrt{2m}) \\ &= O(\alpha' \sigma m q). \end{aligned}$$

The second inequality above follows from Cauchy-Schwartz and the third inequality follows from Lemma 3 (Item 5) and 1. □

Parameter selection. Now, to satisfy the correctness requirement and make the security proof work, we need that

- the error term is less than $q/5$ with overwhelming probability (i.e., $q > \Omega(\alpha' \sigma m q)$),
- that $\mathsf{TrapGen}$ can operate (i.e., $m \ge 6n\lceil \log q \rceil$),
- that the leftover hash lemma (Lemma 4) can be applied in the security proof (i.e., $m = (n+1)\log q + \omega(\log n)$),
- that σ is sufficiently large so that the distribution of private keys in the real world is the same as that in the simulation, (i.e., $\sigma > \sigma_0 = \omega(\sqrt{n \log q \log m})$ and $\sigma > m \cdot (1+\delta) \cdot \omega(\sqrt{\log m})$, where the latter condition turns out to be more restrictive),
- that the ReRand algorithm in the security proof works (i.e., $\alpha'/2\alpha > \sqrt{2} \cdot m(\delta+1)$ and $\alpha q > \omega(\sqrt{\log m})$. See Lemma 2),

– that the worst case to average case reduction works (i.e., $\alpha q > 2\sqrt{2n}$).

To satisfy the above requirements, we set the parameters as follows:

$$\begin{aligned} &m = O(n \log q), && q = n^{7/2} \cdot \delta^2 \cdot \omega(\log^{7/2} n), && \sigma = m \cdot \delta \cdot \omega(\sqrt{\log m}) \\ &\alpha q = 3\sqrt{n}, && \alpha' q = 5\sqrt{n} \cdot m \cdot \delta. && \end{aligned}$$

Here, the parameter δ is determined by the compatible algorithms corresponding to F.

5.6 Security Proof

Theorem 3. *If $\mathsf{F} : \mathcal{K} \times \mathcal{X} \to \{0,1\}$ is a partitioning function and $(\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval})$ are the corresponding δ-compatible algorithms, our scheme achieves adaptively-anonymous security assuming $\mathsf{dLWE}_{n,m+1,q,\alpha}$.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the adaptively-anonymous security of the scheme. In addition, let $\epsilon = \epsilon(\lambda)$ and $Q = Q(\lambda)$ be its advantage and the upper bound of the number of key extraction queries, respectively. By assumption, $Q(\lambda)$ is polynomially bounded and there exists a noticeable function $\epsilon_0(\lambda)$ such that $\epsilon(\lambda) \geq \epsilon_0(\lambda)$ holds for infinitely many λ. By the property of the partitioning function (Definition 7, Item 1), we have that

$$K \in \mathcal{K} \qquad \text{where} \qquad K \overset{\$}{\leftarrow} \mathsf{PrtSmp}(1^\lambda, Q, \epsilon_0)$$

holds with probability 1 for all sufficiently large λ. Here, PrtSmp is the sampling algorithm associated to F. Therefore, in the following, we assume that this condition always holds. We prove the security of the scheme via the following sequence of games. In each game, a value $\mathsf{coin}' \in \{0,1\}$ is defined. While it is set $\mathsf{coin}' = \widehat{\mathsf{coin}}$ in the first game, these values might be different in the later games. In the following, we define E_i to be the event that $\mathsf{coin}' = \mathsf{coin}$.

Game_0 : This is the real security game. Recall that since the ciphertext space is $\mathcal{C} = \mathbb{Z}_q \times \mathbb{Z}_q^{2m}$, in the challenge phase, the challenge ciphertext is set as $\mathsf{ct}^\star = (c_0, \mathbf{c}_1) \overset{\$}{\leftarrow} \mathbb{Z}_q \times \mathbb{Z}_q^{2m}$ if $\mathsf{coin} = 1$. At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By definition, we have

$$\left|\Pr[\mathsf{E}_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon.$$

Game_1 : In this game, we change Game_0 so that the challenger performs the following additional step at the end of the game. First, the challenger runs $\mathsf{PrtSmp}(1^\lambda, Q, \epsilon_0) \to K$ and checks whether the following condition holds:

$$\mathsf{F}(K, \mathsf{ID}^{(1)}) = \cdots = \mathsf{F}(K, \mathsf{ID}^{(Q)}) = 1 \ \wedge \ \mathsf{F}(K, \mathsf{ID}^\star) = 0 \tag{17}$$

where $\mathsf{ID}^\star$ is the challenge identity, and $\mathsf{ID}^{(1)}, \ldots, \mathsf{ID}^{(Q)}$ are identities for which $\mathcal{A}$ has made key extraction queries. If it does not hold, the challenger ignores the output $\widehat{\mathsf{coin}}$ of $\mathcal{A}$, and sets $\mathsf{coin}' \overset{\$}{\leftarrow} \{0,1\}$. In this case, we say that the challenger aborts. If condition (17) holds, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By Lemma 6 and the second property of the partitioning function (Definition 7, Item 2),

$$\left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| \quad \geq \quad \gamma_{\min}\epsilon - \frac{1}{2}(\gamma_{\max} - \gamma_{\min})$$

$$\begin{aligned} &\geq \gamma_{\min}\epsilon_0 - \frac{1}{2}(\gamma_{\max} - \gamma_{\min}) \\ &= \tau \end{aligned}$$

holds for infinitely many λ and a noticeable function $\tau = \tau(\lambda)$, where $\gamma_{\min}$, $\gamma_{\max}$, and τ are specified by ϵ_0, Q, and the underlying partitioning function F.

Game_2 : In this game, we change the way $\mathbf{B}_0$ and $\mathbf{B}_{i,j}$ are chosen. At the beginning of the game, the challenger runs $K \xleftarrow{\$} \mathsf{PrtSmp}(1^\lambda, Q, \epsilon_0)$, computes $\kappa = \mathsf{Encode}(K)$, and samples $\mathbf{R}_0, \mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m\times m}$ for $i \in [u]$. Recall that by our assumption, $K \in \mathcal{K}$ and thus $\kappa \in \{0,1\}^u$. Then, we define $\mathbf{B}_0$ and $\mathbf{B}_i$ as

$$\mathbf{B}_0 = \mathbf{A}\mathbf{R}_0 \qquad \text{and} \qquad \mathbf{B}_i = \mathbf{A}\mathbf{R}_i + \kappa_i \cdot \mathbf{G} \qquad \text{for } i \in [u] \tag{18}$$

where $\kappa_i \in \{0,1\}$ is the i-th bit of κ. The rest of the game is the same as in Game_1.

By Lemma 4, the distributions

$$\left(\ \mathbf{A},\ \mathbf{A}\mathbf{R}_0,\ \{\mathbf{A}\mathbf{R}_i + \kappa_i \cdot \mathbf{G}\}_{i\in[u]}\ \right) \quad \text{and} \quad \left(\ \mathbf{A},\ \mathbf{B}_0,\ \{\mathbf{B}_i\}_{i\in[u]}\ \right)$$

are $\mathsf{negl}(n)$-close, where $\mathbf{B}_0, \mathbf{B}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$. Therefore, we have

$$|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(n).$$

Before describing the next game, we define $\mathbf{R}_{\mathsf{ID}}$ for $\mathsf{ID} \in \{0,1\}^k$ as

$$\mathbf{R}_{\mathsf{ID}} = \mathsf{TrapEval}\left(K, \mathsf{ID}, \mathbf{A}, \{\mathbf{R}_i\}_{i\in[u]}\right).$$

Note that we have

$$\|\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}}\|_\infty \leq \|\mathbf{R}_0\|_\infty + \|\mathbf{R}_{\mathsf{ID}}\|_\infty \leq \delta + 1 \tag{19}$$

where the second inequality follows from $\mathbf{R}_0, \mathbf{R}_i \in \{-1,1\}^{m\times m}$ and the δ-compatibility of $\mathsf{TrapEval}$. Furthermore, from the property of $\mathsf{TrapEval}$, when condition (17) is satisfied, we have

$$\mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}} = \begin{cases} \mathbf{A}\cdot(\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}) & \text{for} \quad \mathsf{ID} = \mathsf{ID}^\star \\ \mathbf{A}\cdot(\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}}) + \mathbf{G} & \text{for} \quad \mathsf{ID} \in \{\mathsf{ID}^{(1)}, \ldots, \mathsf{ID}^{(Q)}\} \end{cases} \tag{20}$$

where $\mathbf{B}_{\mathsf{ID}} = \mathsf{PubEval}\left(\mathsf{ID}, \{\mathbf{B}_i\}_{i\in[u]}\right)$.

Game_3 Recall that in the previous game, the challenger aborts at the end of the game, if condition (17) is not satisfied. In this game, we change the game so that the challenger aborts as soon as the abort condition becomes true. Since this is only a conceptual change, we have

$$\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3].$$

Game_4 In this game, we change the way the matrix $\mathbf{A}$ is sampled. Namely, Game_4 challenger picks $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ instead of generating it with a trapdoor. By Lemma 3, this makes only negligible difference. Furthermore, we also change the way the key extraction queries are answered. When $\mathcal{A}$ makes a key extraction query for an identity ID, the challenger first checks whether $\mathsf{F}(K, \mathsf{ID}) = 0$ and aborts if so (as specified in Game_3). Otherwise, it computes $[\mathbf{A} \| \mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}]_\sigma^{-1} = [\mathbf{A} \| \mathbf{A}(\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}}) + \mathbf{G}]_\sigma^{-1}$ from $\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}}$ using the algorithm in Lemma 3 (Item 3), samples

$$\mathbf{e} \xleftarrow{\$} [\mathbf{A} \| \mathbf{A}(\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}}) + \mathbf{G}]_\sigma^{-1}(\mathbf{u}),$$

and returns it to $\mathcal{A}$. Note that the private key was sampled using $\mathbf{A}_{\sigma_0}^{-1}$ in the previous game. By Eq.(19), (20), and $\sigma > m(\delta+1) \cdot \omega(\sqrt{\log m})$, it follows from Lemma 3 that the above change alters the view of the adversary only negligibly. Thus, we have

$$|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(n).$$

Game_5 : In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to create the challenge ciphertext for the identity $\mathsf{ID}^\star$ and the message M, Game_5 challenger first picks $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $x_0 \xleftarrow{\$} D_{\mathbb{Z},\alpha q}$, $\bar{\mathbf{x}}_1 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha q}$ and sets $w_0 := \mathbf{s}^\top \mathbf{u} + x_0$ and $\mathbf{w}_1^\top := \mathbf{s}^\top \mathbf{A} + \bar{\mathbf{x}}_1^\top$. Then, it computes $\mathbf{R}_{\mathsf{ID}^\star}$ and sets the challenge ciphertext $(c_0, \mathbf{c}_1) \in \mathbb{Z}_q \times \mathbb{Z}_q^{2m}$ as

$$c_0 := w_0 + \mathsf{M} \cdot \lceil q/2 \rceil, \qquad \mathbf{c}_1^\top := \mathsf{ReRand}\left(\mathbf{w}_1, [\mathbf{I}_m \| \mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}], \alpha q, \alpha'/2\alpha\right) \tag{21}$$

where $\mathbf{I}_m$ is the identity matrix with size m.

We show that the view of $\mathcal{A}$ in Game_5 is negligibly close to that in Game_4. To see this, we apply Lemma 2 with $\mathbf{V} := [\mathbf{I}_m \| \mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}]$, $\mathbf{x} := \bar{\mathbf{x}}_1$, and $\mathbf{b}^\top := \mathbf{s}^\top \mathbf{A}$ to obtain that the distribution of $\mathbf{c}_1$ is negligibly close to the following:

$$\begin{aligned} \mathbf{c}_1^\top &= \mathbf{s}^\top \mathbf{A} \, [\mathbf{I}_m \| \mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}] + [\mathbf{x}_1^\top \| \mathbf{x}_2^\top] \\ &= \mathbf{s}^\top \, [\mathbf{A} \| \mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}^\star}] + [\mathbf{x}_1^\top \| \mathbf{x}_2^\top] \end{aligned}$$

where $\mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha' q}$. The second equality follows from Eq.(20). Note that we can apply the lemma because

$$\alpha'/2\alpha > \sqrt{2} \cdot m \cdot (\delta+1) \geq \sqrt{m} \cdot \sqrt{2m} \cdot \|\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}\|_\infty \geq \|\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}\|_2$$

where the second inequality follows from Eq.(19) (with $\mathsf{ID} = \mathsf{ID}^\star$) and the third from the general inequality regarding the relationship between the infinity norm and the operator norm. We may therefore conclude that

$$|\Pr[\mathsf{E}_4] - \Pr[\mathsf{E}_5]| \leq \mathsf{negl}(n).$$

Game_6 In this game, we further change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to create the challenge ciphertext Game_6 challenger first picks $v_0 \xleftarrow{\$} \mathbb{Z}_q$, $\mathbf{v}_1 \xleftarrow{\$} \mathbb{Z}_q^m$, $x_0 \xleftarrow{\$} D_{\mathbb{Z},\alpha q}$, and $\bar{\mathbf{x}}_1 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha q}$ and sets $w_0 := v_0 + x_0$ and $\mathbf{w}_1 := \mathbf{v}_1 + \bar{\mathbf{x}}_1$. Then it computes $\mathbf{R}_{\mathsf{ID}^\star}$ and sets the challenge ciphertext as Eq.(21).

We claim that $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]|$ is negligible assuming $\mathsf{dLWE}_{n,m+1,q,\alpha}$. To show this, we construct an adversary $\mathcal{B}$ against the problem using $\mathcal{A}$, which is described as follows.

$\mathcal{B}$ is given the problem instance of LWE $(\mathbf{A}', \mathbf{w}' = \mathbf{v}' + \bar{\mathbf{x}}) \in \mathbb{Z}_q^{n \times (m+1)} \times \mathbb{Z}_q^{m+1}$ where $\bar{\mathbf{x}} \xleftarrow{\$} D_{\mathbb{Z}^{m+1},\alpha q}$. The task of $\mathcal{B}$ is to distinguish whether $\mathbf{v}'^\top = \mathbf{s}^\top \mathbf{A}'$ for $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ or $\mathbf{v}' \xleftarrow{\$} \mathbb{Z}_q^{m+1}$. Let the first column of $\mathbf{A}'$ be $\mathbf{u} \in \mathbb{Z}_q^n$ and the last m columns be $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. Further, let the first coefficient of $\mathbf{w}'$ be w_0 and the last m coefficients be $\mathbf{w}_1$. Using these terms, $\mathcal{B}$ sets mpk

as Eq.(18). At any point in the game, $\mathcal{B}$ aborts and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$ if condition (17) is not satisfied. During the game, key extraction queries made by $\mathcal{A}$ can be answered as in Game_4 without knowing $\mathbf{A}_{\sigma_0}^{-1}$. To generate the challenge ciphertext, it first picks $\mathsf{coin} \xleftarrow{\$} \{0,1\}$. If $\mathsf{coin} = 0$, it generates the challenge ciphertext as in Eq.(21) using w_0 and $\mathbf{w}_1$, and returns it to $\mathcal{A}$. Note that secret randomness used to generate w_0 and $\mathbf{w}_1$ (namely, $\mathbf{s}$ and $\bar{\mathbf{x}}$) is not necessary to perform this. If $\mathsf{coin} = 1$, $\mathcal{B}$ returns a random ciphertext. At the end of the game, coin' is defined. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

It can be easily seen that if $(\mathbf{A}', \mathbf{w}')$ is a valid LWE sample (i.e., ${\mathbf{v}'}^\top = \mathbf{s}^\top \mathbf{A}'$), the view of the adversary corresponds to Game_5. If $\mathbf{v}' \xleftarrow{\$} \mathbb{Z}_q^{m+1}$, the view of $\mathcal{A}$ corresponds to that of Game_6. It is clear that the advantage of $\mathcal{B}$ is $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]|$. Assuming $\mathsf{dLWE}_{n,m+1,q,\alpha}$, we have

$$|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]| = \mathsf{negl}(n).$$

Game_7 In this game, we further change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to create the challenge ciphertext Game_6 challenger first picks $v_0 \xleftarrow{\$} \mathbb{Z}_q$, $\mathbf{v}_1 \xleftarrow{\$} \mathbb{Z}_q^m$, $x_0 \xleftarrow{\$} D_{\mathbb{Z},\alpha q}$, and $\mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha' q}$ and computes $\mathbf{R}_{\mathsf{ID}^\star}$. Then, it sets the challenge ciphertext as

$$c_0 := v_0 + \mathsf{M} \cdot \lceil q/2 \rceil, \qquad \mathbf{c}_1^\top := \left[\mathbf{v}_1^\top \middle\| \mathbf{v}_1^\top (\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star})\right] + \left[\mathbf{x}_1^\top \middle\| \mathbf{x}_2^\top\right]. \tag{22}$$

Similarly to the change from Game_4 to Game_5, by setting $\mathbf{V} := [\mathbf{I}_m \| \mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}]$, $\mathbf{x} := \bar{\mathbf{x}}_1$, and $\mathbf{b} := \mathbf{v}_1$ (as in Game_6) and applying Lemma 2, it can be seen that this change alters the distribution of the challenge ciphertext only negligibly. Thus, we have

$$|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_7]| = \mathsf{negl}(n).$$

Game_8 In this game, we change the challenge ciphertext to be a random vector, regardless of whether $\mathsf{coin} = 0$ or $\mathsf{coin} = 1$. Namely, Game_8 challenger generates the challenge ciphertext $(c_0, \mathbf{c}_1)$ as $c_0 \xleftarrow{\$} \mathbb{Z}_q$ and $\mathbf{c}_1 \xleftarrow{\$} \mathbb{Z}_q^m$. In this game, the value coin is independent from the view of $\mathcal{A}$. Therefore, $\Pr[\mathsf{E}_8] = 1/2$.

We now proceed to bound $|\Pr[\mathsf{E}_7] - \Pr[\mathsf{E}_8]|$. Since Game_7 and Game_8 differ only in the creation of the challenge ciphertext when $\mathsf{coin} = 0$, we focus on this case. First, it is easy to see that c_0 is uniformly random over $\mathbb{Z}_q$ in both Game_7 and Game_8. We also have to show that the distribution of $\mathbf{c}_1 = \left[\mathbf{v}_1^\top \| \mathbf{v}_1^\top (\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star})\right] + \left[\mathbf{x}_1^\top \| \mathbf{x}_2^\top\right]$ in Game_7 is $\mathsf{negl}(n)$-close to the uniform distribution over $\mathbb{Z}_q^{2m}$. To see this, we first observe that the following distributions are $\mathsf{negl}(n)$-close:

$$(\mathbf{A}, \mathbf{A}\mathbf{R}_0, \mathbf{v}_1^\top, \mathbf{v}_1^\top \mathbf{R}_0) \approx (\mathbf{A}, \mathbf{A}', \mathbf{v}_1^\top, {\mathbf{v}'_1}^\top) \approx (\mathbf{A}, \mathbf{A}\mathbf{R}_0, \mathbf{v}_1^\top, {\mathbf{v}'_1}^\top), \tag{23}$$

where $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{R}_0 \xleftarrow{\$} \{-1,1\}^{m \times m}$, $\mathbf{v}_1, \mathbf{v}'_1 \xleftarrow{\$} \mathbb{Z}_q^m$. It can be seen that the first and the second distributions are $\mathsf{negl}(n)$-close, by applying Lemma 4 for $[\mathbf{A}^\top \| \mathbf{v}_1]^\top \in \mathbb{Z}_q^{(n+1) \times m}$ and $\mathbf{R}_0$. It can also be seen that the second and the third distributions are $\mathsf{negl}(n)$-close, by applying the same lemma for $\mathbf{A}$ and $\mathbf{R}_0$. From the above, we have that the following distributions are statistically close:

$$(\mathbf{A}, \mathbf{A}\mathbf{R}_0, \mathbf{v}_1^\top + \mathbf{x}_1^\top, \mathbf{v}_1^\top(\mathbf{R}_0 + \mathbf{R}_{\mathsf{ID}^\star}) + \mathbf{x}_2^\top) \quad \approx \quad (\mathbf{A}, \mathbf{A}\mathbf{R}_0, \mathbf{v}_1^\top + \mathbf{x}_1^\top, {\mathbf{v}'_1}^\top + \mathbf{v}_1^\top \mathbf{R}_{\mathsf{ID}^\star} + \mathbf{x}_2^\top)$$

$$\approx \quad (\mathbf{A}, \mathbf{A}\mathbf{R}_0, \mathbf{v}_1^\top, {\mathbf{v}'_1}^\top)$$

where $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$, $\mathbf{R}_0 \overset{\$}{\leftarrow} \{-1,1\}^{m \times m}$, $\mathbf{v}_1, \mathbf{v}'_1 \overset{\$}{\leftarrow} \mathbb{Z}_q^m$. The first and the second distributions above are $\mathsf{negl}(n)$-close by Eq.(23), whereas the second and the third distributions are $\mathsf{negl}(n)$-close by the fact that $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{R}_i$, which are used to compute $\mathbf{R}_{\mathsf{ID}^\star}$, are chosen independently random from other variables. Therefore, we may conclude that

$$|\Pr[\mathsf{E}_7] - \Pr[\mathsf{E}_8]| = \mathsf{negl}(n).$$

Analysis. From the above, we have that

$$\begin{aligned}
\left|\Pr[\mathsf{E}_8] - \frac{1}{2}\right| &= \left|\Pr[\mathsf{E}_1] - \frac{1}{2} + \sum_{i=1}^{7} \Pr[\mathsf{E}_{i+1}] - \Pr[\mathsf{E}_i]\right| \\
&\geq \left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| - \sum_{i=1}^{7} |\Pr[\mathsf{E}_{i+1}] - \Pr[\mathsf{E}_i]| \\
&\geq \tau(\lambda) - \mathsf{negl}(\lambda).
\end{aligned}$$

for infinitely many λ. Since $\Pr[\mathsf{E}_8] = 1/2$, we have $\tau(\lambda) \leq \mathsf{negl}(\lambda)$ for infinitely many λ. This is a contradiction since $\tau(\lambda)$ is noticeable. This completes the proof of Theorem 3. □

5.7 Multi-bit Variant

Here, we explain how to extend our scheme to be a multi-bit variant without increasing much the size of the master public keys and ciphertexts following [PVW08, ABB10a, Yam16]. (However, it comes with longer private keys.) To modify the scheme so that it can deal with the message space of length ℓ_M, we replace $\mathbf{u} \in \mathbb{Z}_q^n$ in mpk with $\mathbf{U} \in \mathbb{Z}_q^{n \times \ell_M}$. The component c_0 in the ciphertext is replaced with $\mathbf{c}_0^\top = \mathbf{s}^\top \mathbf{U} + \mathbf{x}_0^\top + \mathsf{M}\lceil q/2 \rceil$, where $\mathbf{x}_0 \overset{\$}{\leftarrow} D_{\mathbb{Z}^{\ell_M}, \alpha q}$ and $\mathsf{M} \in \{0,1\}^{\ell_M}$ is the message to be encrypted. The private key is replaced to be $\mathbf{E} \in \mathbb{Z}^{m \times \ell_M}$, where $\mathbf{E}$ is chosen as $\mathbf{E} \overset{\$}{\leftarrow} [\mathbf{A} \| \mathbf{B}_0 + \mathbf{B}_{\mathsf{ID}}]_\sigma^{-1}(\mathbf{U})$. We can prove security for the multi-bit variant from $\mathsf{dLWE}_{n, m+\ell_M, q, \alpha}$ by naturally extending the proof of Theorem 3. We note that the same parameters as in Sec. 5.5 will also work for the multi-bit variant. By this change, the sizes of the master public keys, ciphertexts, and private keys become $\tilde{O}(n^2 u + n\ell_M)$, $\tilde{O}(n + \ell_M)$, and $\tilde{O}(n\ell_M)$ from $\tilde{O}(n^2 u)$, $\tilde{O}(n)$, and $\tilde{O}(n)$, respectively. The sizes of the master public keys and ciphertexts will be asymptotically the same as long as $\ell_M = \tilde{O}(n)$. To deal with longer messages, we employ a KEM-DEM approach as suggested in [Yam16]. Namely, we encrypt a random ephemeral key of sufficient length and then encrypt the message by using the ephemeral key.

6 Our VRF Scheme Based on $\mathsf{F}_{\mathsf{MAH}}$

6.1 Construction

Here, we construct a verifiable random function scheme based on the partitioning function $\mathsf{F}_{\mathsf{MAH}}$. We let the input and output space of the scheme be $\mathcal{X} = \{0,1\}^k$ and $\mathcal{Y} = \mathbb{G}_T$, respectively. Let $\eta := \eta(\lambda)$, $\ell := \ell(\lambda)$, $C : \{0,1\}^k \to \{0,1\}^\ell$, and S be as in Sec. 4.3. We also introduce $\ell_1 := \ell_1(\lambda)$ and $\ell_2 = \ell_2(\lambda)$ such that $\ell = \ell_1 \ell_2$. These parameters will control the trade-offs between sizes of proofs and verification keys. A typical choice would be $(\ell_1, \ell_2) = (O(\sqrt{\ell}), O(\sqrt{\ell}))$ or $(\ell_1, \ell_2) = (O(\ell), O(1))$.

$\mathsf{Gen}(1^\lambda)$: On input 1^λ, it chooses a group description $\Pi \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$. It chooses random generators $g, h \xleftarrow{\$} \mathbb{G}^*$ and $w_1, \ldots, w_\eta \xleftarrow{\$} \mathbb{Z}_p$. It then outputs

$$\mathsf{vk} = \left(\Pi,\ g,\ h,\ \left\{ W_{i,j_1} := g^{w_i^{j_1}} \right\}_{(i,j_1)\in[\eta]\times[\ell_1]} \right) \qquad \text{and} \qquad \mathsf{sk} = \left(\{w_i\}_{i\in[\eta]} \right).$$

$\mathsf{Eval}(\mathsf{sk}, X)$: Given $X \in \{0,1\}^k$, it first computes $\mathsf{S}(X) = \{s_1, \ldots, s_\ell\} \subset [2\ell]$,

$$\theta = \prod_{(i,j)\in[\eta]\times[\ell]} (w_i + s_j), \qquad \text{and} \qquad \theta_{i,j_2} = \prod_{(i',j')\in\Omega_{i,j_2}} (w_{i'} + s_{j'}) \tag{24}$$

for $(i, j_2) \in [\eta] \times [\ell_2]$, where

$$\Omega_{i,j_2} = \left\{ \quad (i', j') \in [\eta] \times [\ell] \quad \middle| \quad (\ i' \in [i-1]\ ) \ \vee\ (\ i' = i\ \wedge\ j' \in [j_2\ell_1]\ ) \quad \right\}.$$

We note that $\theta = \theta_{\eta,\ell_2}$. If $\theta \equiv 0 \mod p$, it outputs $Y = 1_{\mathbb{G}_T}$ and $\pi = (\{\pi_{i,j_2} = 1_{\mathbb{G}}\}_{(i,j_2)\in[\eta]\times[\ell_2]})$[§]. Otherwise, it outputs

$$Y = e(g,h)^{1/\theta} \qquad \text{and} \qquad \pi = \left(\left\{ \pi_{i,j_2} = g^{1/\theta_{i,j_2}} \right\}_{(i,j_2)\in[\eta]\times[\ell_2]} \right).$$

$\mathsf{Verify}(\mathsf{vk}, X, Y, \pi)$: It first checks the validity of vk by the following steps. It outputs 0 if any of the following does not hold:

1. vk is of the form $(\Pi, g, h, \{W_{i,j_1}\}_{(i,j_1)\in[\eta]\times[\ell_1]})$.
2. $\mathsf{GrpVfy}(\Pi) \to 1$, $g, h \in \mathbb{G}^*$, and $W_{i,j_1} \in \mathbb{G}$ for all $(i, j_1) \in [\eta] \times [\ell_1]$.
3. $e(W_{i,1}, W_{i,j_1-1}) = e(g, W_{i,j_1})$ for all $(i, j_1) \in [\eta] \times [2, \ell_1]$.

It then checks the validity of Y and π. To do this, it computes $\Phi_{i,j_2} \in \mathbb{G}$ for $(i, j_2) \in [\eta] \times [\ell_2]$ as

$$\Phi_{i,j_2} := g^{\varphi_{j_2,0}} \cdot \prod_{j_1\in[\ell_1]} W_{i,j_1}^{\varphi_{j_2,j_1}}, \tag{25}$$

where $\{\varphi_{j_2,j_1} \in \mathbb{Z}_p\}_{(j_2,j_1)\in[\ell_2]\times[0,\ell_1]}$ are the coefficients of the following polynomial:

$$\prod_{j'\in[(j_2-1)\ell_1+1, j_2\ell_1]} \left(\mathsf{Z} + s_{j'}\right) = \varphi_{j_2,0} + \sum_{j_1\in[\ell_1]} \varphi_{j_2,j_1} \mathsf{Z}^{j_1} \in \mathbb{Z}_p[\mathsf{Z}].$$

It outputs 0 if any of the following does not hold:

4. $X \in \{0,1\}^k$, $Y \in \mathbb{G}_T$, π is of the form $\pi = (\{\pi_{i,j_2} \in \mathbb{G}\}_{(i,j_2)\in[\eta]\times[\ell_2]})$.
5. If there exists $(i, j_2) \in [\eta] \times [\ell_2]$ such that $\Phi_{i,j_2} = 1_{\mathbb{G}}$, we have $Y = 1_{\mathbb{G}_T}$ and $\pi_{i,j_2} = 1_{\mathbb{G}}$ for all $(i, j_2) \in [\eta] \times [\ell_2]$.
6. If $\Phi_{i,j_2} \neq 1_{\mathbb{G}}$ for all $(i, j_2) \in [\eta] \times [\ell_2]$, the following equation holds for all $(i, j_2) \in [\eta] \times [\ell_2]$:

$$e(\pi_{i,j_2}, \Phi_{i,j_2}) = e(\pi_{i,j_2-1}, g) \tag{26}$$

   where we define $\pi_{i,0} := \pi_{i-1,\ell_2}$ for $i \geq 2$ and $\pi_{1,0} := g$.
7. $e(\pi_{\eta,\ell_2}, h) = Y$ holds.

If all the above conditions hold, it outputs 1.

[§] The event occurs with only negligible probability. This choice of the output is arbitrary and can be replaced with any fixed group elements.

6.2 Correctness, Unique Provability, and Pseudorandomness of the Scheme

Correctness. We prove correctness of the scheme. Let us assume that (vk, Y, π) are correctly generated as $(\mathsf{vk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ and $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$. It is straightforward to see that it passes Step 1, 2, 3, and 4 in the verification algorithm. In the case where there exists $(i, j_2) \in [\eta] \times [\ell_2]$ such that $\log_g \Phi_{i,j_2} \equiv 0 \mod p$, we have $\theta \equiv 0 \mod p$ because $\theta \equiv \prod_{(i,j_2)\in[\eta]\times[\ell_2]} \log_g \Phi_{i,j_2} \mod p$. Then, it is straightforward to see that honestly generated (vk, Y, π) passes Step 5, 6, and 7. Next, we proceed to consider the case where $\log_g \Phi_{i,j_2} \not\equiv 0 \mod p$ for all $(i, j_2) \in [\eta] \times [\ell_2]$. First, it trivially passes Step 5. Furthermore, since $\theta = \theta_{\eta,\ell_2}$, it also passes Step 7. Finally, we show that it passes Step 6 by proving Eq.(26). Let us denote $\theta_{i,0} := \theta_{i-1,\ell_2}$, $\Omega_{i,0} := \Omega_{i-1,\ell_2}$ for $i \geq 2$, $\theta_{1,0} := 1$, and $\Omega_{1,0}$ be the empty set. We have

$$\begin{aligned}
\log_g \Phi_{i,j_2} &= \varphi_{j_2,0} + \sum_{j_1\in[\ell_1]} \varphi_{j_2,j_1} w_i^{j_1} \\
&= \prod_{j'\in[(j_2-1)\ell_1+1, j_2\ell_1]} (w_i + s_{j'}) \\
&= \left(\prod_{(i',j')\in\Omega_{i,j_2}} (w_{i'} + s_{j'}) \right) \Bigg/ \left(\prod_{(i',j')\in\Omega_{i,j_2-1}} (w_{i'} + s_{j'}) \right) \\
&= \theta_{i,j_2}/\theta_{i,j_2-1}. \qquad (27)
\end{aligned}$$

This immediately implies Eq.(26). This completes the proof of the correctness.

Unique Provability. We prove the unique provability of the scheme. We have to show that for any $\mathsf{vk} \in \{0,1\}^*$, there does not exist any (Y, π, Y', π') such that $\mathsf{Verify}(\mathsf{vk}, X, Y, \pi) = \mathsf{Verify}(\mathsf{vk}, X, Y', \pi') = 1$ and $Y \neq Y'$.

- Step 1 checks whether Π contains valid certified bilinear group parameters. Step 2 checks whether $g \in \mathbb{G}^*$, $h \in \mathbb{G}^*$, and $W_{i,j_1} \in \mathbb{G}$. Thus, we may assume in the following that all these group elements are valid and have a unique encoding. In particular, there exists unique $w_i \in \mathbb{Z}_p$ such that $g^{w_i} = W_{i,1}$ for all $i \in [\eta]$. Step 3 verifies whether $W_{i,j_1} \overset{?}{=} g^{w_i^{j_1}}$ for all $(i, j_1) \in [\eta] \times [2, \ell_1]$. We may assume that all these conditions are satisfied in the following. At this point, θ, θ_{i,j_2}, and Φ_{i,j_2} for $(i, j_2) \in [\eta] \times [\ell_2]$ defined as Eq.(24) and (25) are well-defined.

- In Step 4, it is checked whether $X \in \{0,1\}^k$, $Y \in \mathbb{G}_T$, and π is of the form $\pi = (\{\pi_{i,j_2} \in \mathbb{G}\}_{(i,j_2)\in[\eta]\times[\ell_2]})$. We first consider the case where $\log_g \Phi_{i,j_2} \not\equiv 0 \mod p$ holds for all $(i, j_2) \in [\eta] \times [\ell_2]$. In this case, by Eq.(27) and induction, we can see that (Y, π) will pass Step 6 only if $\pi_{i,j_2} = g^{1/\theta_{i,j_2}}$ holds for all $(i, j_2) \in [\eta] \times [\ell_2]$. Furthermore, it will pass Step 7 only if $Y = e(g^{1/\theta_{\eta,\ell_2}}, h) = e(g, h)^{1/\theta}$. Due to the fact that the bilinear group is certified, which guarantees that each group element has a unique encoding and that the bilinear map is non-degenerate, the unique provability follows. Next we consider the case where there exists $(i, j_2) \in [\eta] \times [\ell_2]$ such that $\log_g \Phi_{i,j_2} \equiv 0 \mod p$. In such a case, the only (Y, π) that is accepted by (Step 5 of) the verification algorithm is $Y = 1_{\mathbb{G}_T}$ and $\pi = \{\pi_{i,j_2} = 1_{\mathbb{G}}\}_{(i,j_2)\in[\eta]\times[\ell_2]}$. Similarly to the above case, the fact that the bilinear group is certified implies the unique provability.

This completes the proof of the unique provability.

Pseudorandomness. The following theorem asserts the pseudorandomness of the scheme.

Theorem 4. *Our scheme satisfies pseudorandomness assuming L-DDH with $L = (4\ell+1)\eta + \ell_1$.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks pseudorandomness of the scheme. In addition, let $\epsilon = \epsilon(\lambda)$ and $Q = Q(\lambda)$ be its advantage and the upper bound on the number of evaluation queries, respectively. By assumption, $Q(\lambda)$ is polynomially bounded and there exists a noticeable function $\epsilon_0(\lambda)$ such that $\epsilon(\lambda) \geq \epsilon_0(\lambda)$ holds for infinitely many λ. By the property of the partitioning function (Definition 7, Item 1), we have that

$$|\mathsf{T}| < \eta \qquad \text{where} \qquad \mathsf{T} \overset{\$}{\leftarrow} \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q, \epsilon_0)$$

holds with probability 1 for all sufficiently large λ. Therefore, in the following, we assume that this condition always holds. We show the security of the scheme via the following sequence of games. In each game, a value $\mathsf{coin}' \in \{0,1\}$ is defined. While it is set $\mathsf{coin}' = \widehat{\mathsf{coin}}$ in the first game, these values might be different in the later games. In the following, we define E_i be the event that $\mathsf{coin}' = \mathsf{coin}$.

Game_0 : This is the real security game. Recall that since the range of the function is $\mathcal{Y} = \mathbb{G}_T$, in the challenge phase, $Y_1^\star \overset{\$}{\leftarrow} \mathbb{G}_T$ is returned to $\mathcal{A}$ if $\mathsf{coin} = 1$. At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By definition, we have

$$\left|\Pr[\mathsf{E}_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon.$$

Game_1 : In this game, we change Game_0 so that the challenger performs the following additional step at the end of the game. First, the challenger runs $\mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q, \epsilon_0) \to \mathsf{T} \subseteq [2\ell]$ and checks whether the following condition holds:

$$\mathsf{T} \not\subseteq \mathsf{S}(X^{(1)}) \ \wedge \ \cdots \ \wedge \ \mathsf{T} \not\subseteq \mathsf{S}(X^{(Q)}) \ \wedge \ \mathsf{T} \subseteq \mathsf{S}(X^\star) \tag{28}$$

where $X^\star$ is chosen by $\mathcal{A}$ at the challenge phase, and $X^{(1)}, \ldots, X^{(Q)}$ are inputs to the VRF for which $\mathcal{A}$ has queried the evaluation of the function. If it does not hold, the challenger ignores the output $\widehat{\mathsf{coin}}$ of $\mathcal{A}$, and sets $\mathsf{coin}' \overset{\$}{\leftarrow} \{0,1\}$. In this case, we say that the challenger aborts. If condition (28) holds, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By Lemma 6 and 7 (See also Definition 7, Item 2),

$$\begin{aligned} \left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| &\geq \gamma_{\min}\epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2} \\ &\geq \gamma_{\min}\epsilon_0 - \frac{\gamma_{\max} - \gamma_{\min}}{2} \\ &= \tau \end{aligned}$$

holds for infinitely many λ and a noticeable function $\tau = \tau(\lambda)$. Here, $\gamma_{\min}$, $\gamma_{\max}$, and τ are specified by ϵ_0, Q, and the underlying partitioning function $\mathsf{F}_{\mathsf{MAH}}$.

Game_2 : In this game, we change the way w_i are chosen. At the beginning of the game, the challenger picks $\mathsf{T} \overset{\$}{\leftarrow} \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q, \epsilon_0)$ and parses it as $\mathsf{T} = \{t_1, \ldots, t_{\eta'}\} \subset [2\ell]$. Recall that by our assumption, we have $\eta' < \eta$. It then sets $t_i := 0$ for $i \in [\eta'+1, \eta]$. It then samples $\alpha \overset{\$}{\leftarrow} \mathbb{Z}_p^*$, and $\tilde{w}_i \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ for $i \in [\eta]$. Then, w_i are defined as

$$w_i = \tilde{w}_i \cdot \alpha - t_i \qquad \text{for } i \in [\eta].$$

The rest of the game is the same as in Game_1. The statistical distance of the distributions of $\{w_i\}_{i\in[\eta]}$ in Game_1 and Game_2 is at most η/p, which is negligible. Therefore, we have

$$|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(\lambda).$$

Before describing the next game, for any $\Omega \subseteq [\eta] \times [\ell]$, $\mathsf{T} \subset [2\ell]$ with $|\mathsf{T}| = \eta' < \eta$, and $X \in \{0,1\}^k$, we define polynomials $\mathsf{P}_{X,\Omega}(\mathsf{Z}), \mathsf{Q}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ as

$$\mathsf{P}_{X,\Omega}(\mathsf{Z}) = \prod_{(i,j)\in\Omega} (\tilde{w}_i\mathsf{Z} - t_i + s_j) \qquad \text{and} \qquad \mathsf{Q}(\mathsf{Z}) = \mathsf{Z}^{\eta'-1} \cdot \prod_{(i,j)\in[\eta]\times[-2\ell,2\ell]\setminus\{0\}} (\tilde{w}_i\mathsf{Z} + j)\,,$$

where $\{s_j\}_{j\in[\ell]} = \mathsf{S}(X)$ and $\{t_i\}_{i\in[\eta]}$ are defined as in Game_2 (namely, $\mathsf{T} = \{t_i\}_{i\in[\eta']}$ and $t_i = 0$ for $i > \eta'$). In the special case of $\Omega = [\eta] \times [\ell]$, we denote $\mathsf{P}_X(\mathsf{Z}) := \mathsf{P}_{X,[\eta]\times[\ell]}(\mathsf{Z})$. We state the following lemma, which plays an important roll in our security proof.

Lemma 14. *There exist $\xi_X \in \mathbb{Z}_p^*$ and $\mathsf{R}_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that*

$$\frac{\mathsf{Q}(\mathsf{Z})}{\mathsf{P}_X(\mathsf{Z})} = \begin{cases} \dfrac{\xi_X}{\mathsf{Z}} + \mathsf{R}_X(\mathsf{Z}) & \textit{if} \quad \mathsf{T} \subseteq \mathsf{S}(X) \\ \mathsf{R}_X(\mathsf{Z}) & \textit{if} \quad \mathsf{T} \not\subseteq \mathsf{S}(X) \end{cases}. \tag{29}$$

From the above lemma, we can see that for any $\Omega \subseteq [\eta] \times [\ell]$, it holds that

$$\mathsf{P}_{X,\Omega}(\mathsf{Z}) \mid \mathsf{Q}(\mathsf{Z}) \qquad \text{if} \qquad \mathsf{T} \not\subseteq \mathsf{S}(X),$$

because $\mathsf{P}_{X,\Omega}(\mathsf{Z}) \mid \mathsf{P}_X(\mathsf{Z})$. So as not to interrupt the proof of Theorem 4, we intentionally skip the proof of Lemma 14 for the time being.

Game_3 Recall that in the previous game, the challenger aborts at the end of the game, if condition (28) is not satisfied. In this game, we change the game so that the challenger aborts as soon as the abort condition becomes true. Since this is only a conceptual change, we have

$$\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3].$$

Game_4 In this game, we change the way g is sampled. Namely, Game_4 challenger first picks α and $\tilde{w}_i$ as specified in Game_2. It further picks $\hat{g} \overset{\$}{\leftarrow} \mathbb{G}^*$. Then, it computes (coefficients of) $\mathsf{Q}(\mathsf{Z})$ and sets

$$g := \hat{g}^{\mathsf{Q}(\alpha)}, \qquad W_{i,j_1} = g^{w_i^{j_1}} = \hat{g}^{\mathsf{Q}(\alpha)\cdot(\tilde{w}_i\alpha - t_i)^{j_1}} \qquad \text{for} \qquad (i, j_1) \in [\eta] \times [\ell_1]. \tag{30}$$

It aborts and outputs a random bit if $g = 1_{\mathbb{G}} \Leftrightarrow \mathsf{Q}(\alpha) \equiv 0 \mod p$. It can be seen that the distribution of g and W_{i,j_1} is unchanged, unless $\mathsf{Q}(\alpha) \equiv 0 \mod p$. Since $\mathsf{Q}(\mathsf{Z})$ is a non-zero polynomial with degree $(4\eta\ell + \eta' - 1)$ and α is chosen uniformly at random from $\mathbb{Z}_p^*$, it follows from the Schwartz-Zippel lemma that this happens with probability at most $(4\eta\ell + \eta' - 1)/(p - 1) = \mathsf{negl}(\lambda)$. We therefore have

$$|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(\lambda).$$

Game_5 In this game, we change the way the evaluation queries are answered. By the change introduced in Game_4, we assume $\mathsf{Q}(\alpha) \not\equiv 0 \mod p$ in the following. When $\mathcal{A}$ makes a query for an input X, the challenger first checks whether $\mathsf{T} \subseteq \mathsf{S}(X)$ and aborts if so (as specified in Game_3). Otherwise, it computes $\mathsf{R}_{X,\Omega_{i,j_2}}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{Q}(\mathsf{Z}) = \mathsf{P}_{X,\Omega_{i,j_2}}(\mathsf{Z}) \cdot \mathsf{R}_{X,\Omega_{i,j_2}}(\mathsf{Z})$ for $(i, j_2) \in [\eta] \times [\ell_2]$. Note that such polynomials exist by Lemma 14. Then, it returns

$$Y = e\left(\hat{g}^{\mathsf{R}_{X,\Omega_{\eta,\ell_2}}(\alpha)}, h\right), \qquad \pi = \left(\left\{ \pi_{i,j_2} = \hat{g}^{\mathsf{R}_{X,\Omega_{i,j_2}}(\alpha)} \right\}_{(i,j_2) \in [\eta] \times [\ell_2]} \right) \tag{31}$$

to $\mathcal{A}$. We claim that this is only a conceptual change. To see this, we first observe that

$$\mathsf{P}_{X,\Omega_{i,j_2}}(\alpha) = \prod_{(i',j') \in \Omega_{i,j_2}} \left(\tilde{w}_{i'}\alpha - t_{i'} + s_{j'}\right) = \prod_{(i',j') \in \Omega_{i,j_2}} \left(w_{i'} + s_{j'}\right) = \theta_{i,j_2}. \tag{32}$$

We can see that $\theta_{i,j_2} \not\equiv 0 \mod p$, since otherwise we have $\mathsf{Q}(\alpha) \equiv \mathsf{P}_{X,\Omega_{i,j_2}}(\alpha) \cdot \mathsf{R}_{X,\Omega_{i,j_2}}(\alpha) \equiv \theta_{i,j_2} \cdot \mathsf{R}_{X,\Omega_{i,j_2}}(\alpha) \equiv 0 \mod p$, which is a contradiction. Thus, we have

$$\hat{g}^{\mathsf{R}_{X,\Omega_{i,j_2}}(\alpha)} = \hat{g}^{\mathsf{Q}(\alpha)/\mathsf{P}_{X,\Omega_{i,j_2}}(\alpha)} = g^{1/\mathsf{P}_{X,\Omega_{i,j_2}}(\alpha)} = g^{1/\theta_{i,j_2}}.$$

This indicates that the simulation by the challenger is perfect. Since the view of $\mathcal{A}$ is unchanged, we have

$$\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5].$$

Game_6 : In this game, we change the way the challenge value $Y_0^\star = \mathsf{Eval}(\mathsf{sk}, X^\star)$ is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to generate $Y_0^\star$, it first computes $\xi_{X^\star} \in \mathbb{Z}_p^*$ and $\mathsf{R}_{X^\star}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{Q}(\mathsf{Z})/\mathsf{P}_{X^\star}(\mathsf{Z}) = \xi_{X^\star}/\mathsf{Z} + \mathsf{R}_{X^\star}(\mathsf{Z})$. Note that such $\xi_{X^\star}$ and $\mathsf{R}_{X^\star}(\mathsf{Z})$ exist by Lemma 14 whenever $\mathsf{T} \subseteq \mathsf{S}(X^\star)$. It then sets

$$Y_0^\star = \left(e\left(\hat{g}, h\right)^{1/\alpha}\right)^{\xi_{X^\star}} \cdot e\left(\hat{g}^{\mathsf{R}_{X^\star}(\alpha)}, h\right)$$

and returns it to $\mathcal{A}$. We claim that this is only a conceptual change. This can be seen by observing that

$$e\left(\hat{g}^{1/\alpha}, h\right)^{\xi_{X^\star}} \cdot e\left(\hat{g}^{\mathsf{R}_{X^\star}(\alpha)}, h\right) = e\left(\hat{g}^{\xi_{X^\star}/\alpha + \mathsf{R}_{X^\star}(\alpha)}, h\right) = e\left(\hat{g}^{\mathsf{Q}(\alpha)/\mathsf{P}_{X^\star}(\alpha)}, h\right) = e\left(g, h\right)^{1/\mathsf{P}_{X^\star}(\alpha)}$$

and $\mathsf{P}_{X^\star}(\alpha) = \theta_{\eta,\ell_2}$, where the latter follows from Eq.(32). Since the view of $\mathcal{A}$ is unchanged, we therefore conclude that

$$\Pr[\mathsf{E}_5] = \Pr[\mathsf{E}_6].$$

Game_7 In this game, we change the challenge value to be a random element in $\mathbb{G}_T$ regardless of whether $\mathsf{coin} = 0$ or $\mathsf{coin} = 1$. Namely, Game_7 challenger sets $Y_0^\star \xleftarrow{\$} \mathbb{G}_T$. In this game, the value coin is independent from the view of $\mathcal{A}$. Therefore, $\Pr[\mathsf{E}_7] = 1/2$.

We claim that $|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_7]|$ is negligible assuming L-DDH with $L = (4\ell + 1)\eta + \ell_1$. To show this, we construct an adversary $\mathcal{B}$ against the problem using $\mathcal{A}$, which is described as follows.

$\mathcal{B}$ is given the problem instance $(\Pi, \hat{g}, h, \{\hat{g}^{\alpha^i}\}_{i \in [L]}, \Psi)$ of L-DDH where $\Psi = e(\hat{g}, h)^{1/\alpha}$ or $\Psi \xleftarrow{\$} \mathbb{G}_T$. At any point in the game, $\mathcal{B}$ aborts and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$ if condition (28) is not satisfied. It first sets g and W_{i,j_1} as Eq.(30) and returns $\mathsf{vk} = (\Pi, g, h, \{W_{i,j_1}\}_{(i,j_1) \in [\eta] \times [\ell_1]})$ to $\mathcal{A}$. These terms can be efficiently computable from the problem instance because $\log_{\hat{g}} g$ and $\log_{\hat{g}} W_{i,j_1}$ can be written as polynomials in α with degree at most $\eta' - 1 + 4\eta\ell + \ell_1 < L$ and the coefficients of the polynomials can be efficiently computable. When $\mathcal{A}$ makes an evaluation query on input X, it computes (Y, π) as Eq.(31) and returns it to $\mathcal{A}$. Again, these terms can be efficiently computable from the problem instance, because the degree of $\mathsf{R}_{X, \Omega_{i,j_2}}(\alpha)$ is at most L and coefficients of them can be efficiently computable. When $\mathcal{A}$ makes the challenge query on input $X^\star$, $\mathcal{B}$ first picks $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and returns $Y^\star \xleftarrow{\$} \mathbb{G}$ if $\mathsf{coin} = 1$. Otherwise, it returns

$$Y^\star = \Psi^{\xi_{X^\star}} \cdot e\left(\hat{g}^{\mathsf{R}_{X^\star}(\alpha)}, h\right)$$

to $\mathcal{A}$. Note that $\hat{g}^{\mathsf{R}_{X^\star}(\alpha)}$ can be efficiently computed from the problem instance because the degree of $\mathsf{R}_{X^\star}(\mathsf{Z})$ is at most L. At the end of the game, coin' is defined. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

It can easily be seen that the view of $\mathcal{A}$ corresponds to that of Game_6 if $\Psi = e(\hat{g}, h)^{1/\alpha}$ and Game_7 if $\Psi \xleftarrow{\$} \mathbb{G}_T$. It is clear that the advantage of $\mathcal{B}$ is $|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_7]|$. Assuming L-DDH, we have

$$|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_7]| = \mathsf{negl}(\lambda).$$

Analysis. From the above, we have

$$\begin{aligned}
\left|\Pr[\mathsf{E}_7] - \frac{1}{2}\right| &= \left|\Pr[\mathsf{E}_1] - \frac{1}{2} + \sum_{i=1}^{6} \Pr[\mathsf{E}_{i+1}] - \Pr[\mathsf{E}_i]\right| \\
&\geq \left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| - \sum_{i=1}^{6} |\Pr[\mathsf{E}_{i+1}] - \Pr[\mathsf{E}_i]| \\
&\geq \tau(\lambda) - \mathsf{negl}(\lambda).
\end{aligned}$$

for infinitely many λ. Since $\Pr[\mathsf{E}_7] = 1/2$, this implies $\tau(\lambda) \leq \mathsf{negl}(\lambda)$ for infinitely many λ, which is a contradiction. □

To complete the proof of Theorem 4, it remains to prove Lemma 14.

Proof of Lemma 14. By definition, we have

$$\mathsf{P}_X(\mathsf{Z}) = \prod_{(i,j) \in [\eta] \times [\ell]} (\tilde{w}_i \mathsf{Z} - t_i + s_j).$$

We define $\mathsf{P}_{X,i}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ and $\mathsf{Q}_i(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ for $i \in [\eta]$ as

$$\mathsf{P}_{X,i}(\mathsf{Z}) = \prod_{j \in [\ell]} (\tilde{w}_i \mathsf{Z} - t_i + s_j) \qquad \text{and} \qquad \mathsf{Q}_i(\mathsf{Z}) = \prod_{j \in [-2\ell, 2\ell] \setminus \{0\}} (\tilde{w}_i \mathsf{Z} + j).$$

We have $|-t_i + s_j| \le 2\ell$ since $t_i, s_j \in [2\ell]$. Furthermore, if $t_i \notin \mathsf{S}(X)$, then $-t_i + s_j \ne 0$ for all $j \in [\ell]$. Otherwise, if $t_i \in \mathsf{S}(X)$, there is a unique $j' \in [\ell]$ such that $-t_i + s_{j'} = 0$. These facts together with $\tilde{w}_i \ne 0$ imply that

$$\begin{cases} \mathsf{P}_{X,i}(\mathsf{Z}) \mid \mathsf{Z} \cdot \mathsf{Q}_i(\mathsf{Z}) & \text{if} \quad t_i \in \mathsf{S}(X) \\ \mathsf{P}_{X,i}(\mathsf{Z}) \mid \mathsf{Q}_i(\mathsf{Z}) & \text{if} \quad t_i \notin \mathsf{S}(X) \end{cases}.$$

Let us define $I := \{\ i \in [\eta] \mid t_i \in \mathsf{S}(X)\ \}$ and $\eta'' := |I|$. From the above observation and by the fact that $\mathsf{P}_X(\mathsf{Z}) = \prod_{i\in[\eta]} \mathsf{P}_{X,i}(\mathsf{Z})$, we have

$$\mathsf{P}_X(\mathsf{Z}) \mid \mathsf{Z}^{\eta''} \cdot \prod_{i\in[\eta]} \mathsf{Q}_i(\mathsf{Z}).$$

Observe that $\mathsf{Q}(\mathsf{Z}) = \mathsf{Z}^{\eta'-1} \cdot \prod_{i\in[\eta]} \mathsf{Q}_i(\mathsf{Z})$. If $\mathsf{T} \not\subseteq \mathsf{S}(X)$, we have $\eta'' < \eta'$ and thus $\mathsf{P}_X(\mathsf{Z}) \mid \mathsf{Q}(\mathsf{Z})$. On the other hand, if $\mathsf{T} \subseteq \mathsf{S}(X)$, we have $\eta'' = \eta'$ and thus $\mathsf{P}_X(\mathsf{Z}) \mid \mathsf{Z} \cdot \mathsf{Q}(\mathsf{Z})$. Furthermore, we have $\mathsf{P}_X(\mathsf{Z}) \nmid \mathsf{Q}(\mathsf{Z})$, since $\mathsf{P}_X(\mathsf{Z})$ can be divided by Z exactly η' times whereas $\mathsf{Q}(\mathsf{Z})$ can only be divided by Z exactly $\eta' - 1$ times. The facts that $\mathsf{P}_X(\mathsf{Z}) \nmid \mathsf{Q}(\mathsf{Z})$ and $\mathsf{P}_X(\mathsf{Z}) \mid \mathsf{Z} \cdot \mathsf{Q}(\mathsf{Z})$ imply that there exist $\mathsf{R}_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ and $\xi_X \in \mathbb{Z}_p^*$ satisfying $\mathsf{Q}(\mathsf{Z})/\mathsf{P}_X(\mathsf{Z}) = \xi_X/\mathsf{Z} + \mathsf{R}_X(\mathsf{Z})$. This completes the proof of Lemma 14. □

6.3 A Variant with Short Verification Keys

Here, we introduce a variant of our scheme in Sec. 6.1. In the variant, we remove $\{W_{i,j_1} = g^{w_i^{j_1}}\}_{(i,j_1)\in[\eta]\times[2,\ell_1]}$ from vk. Instead, we add these components to π. We do not change the verification algorithm and other parts of the scheme. It is straightforward to see that the correctness and pseudorandomness of the scheme can still be proven. To prove the unique provability, we observe that the only possible strategy to break is to include invalid $\{W_{i,j_1}\}_{(i,j_1)\in[\eta]\times[2,\ell_1]}$ in the proof. This is because if these values are correct, the unique provability of the original scheme immediately implies that of the modified scheme. However, this strategy does not work since the invalid values will be detected at Step 3 of the verification algorithm using $\{W_{i,1} = g^{w_i}\}_{i\in[\eta]}$ in vk. The advantage of the variant is that the size of vk is small. In particular, vk only consists of $\eta + 2$ group elements in this variant, whereas $\eta\ell_1 + 2$ group elements were required in the scheme in Sec. 6.1. Of course, this change increases the size of the proofs π. The number of group elements will become $\eta(\ell_1 + \ell_2 - 1)$ from $\eta\ell_2$ by this modification. To minimize the size of the proofs we choose $\ell_1 = \ell_2 = \sqrt{\ell}$.

7 Comparisons

Here, we compare our proposed schemes with previous schemes.

New Lattice IBE Schemes. In Sec. 5.4, we showed how to construct an IBE scheme from a partitioning function with associating compatible algorithms. We have two ways of instantiating the scheme.

- By using the partitioning function $\mathsf{F}_{\mathsf{MAH}}$ in Sec. 4.3 and the corresponding compatible algorithms in Sec. 5.2, we obtain our first IBE scheme. The master public key of the scheme only consists of $\omega(\log^2 \lambda)$ matrices.

- By using the partitioning function $\mathsf{F}_{\mathsf{AFF}}$ in Sec. 4.4 and the corresponding compatible algorithms in Sec. 5.3, we obtain our second IBE scheme. The master public key of the scheme is even shorter: It only consists of $\omega(\log \lambda)$ matrices.

Both our schemes achieve the best asymptotic space efficiency (namely, the sizes of the master public keys, ciphertexts, and private keys) among existing IBE schemes that are adaptively secure against *unbounded collusion without sub-exponential security assumptions*. In Table 1 in Sec. 1, we compare our schemes with previous schemes. Note that the scheme by Zhang et al. [ZCZ16] achieves shorter master public key size than ours, but only achieves Q-bounded security. This restriction cannot be removed by just making Q super-polynomial, since the encryption algorithm of the scheme runs in time proportional to Q.

Finally, we note that there are two drawbacks that are common in our schemes. The first drawback is that the encryption algorithm is heavy. Our first scheme requires $\tilde{O}(\lambda)$ times of matrix multiplications for the encryption algorithm. Our second scheme requires even heavier computation. It first computes the description of the "division in $\mathbf{NC}^1$ circuit" [BCH86] and then invokes Barrington's theorem [Bar89] to convert it into a branching program. The second drawback is that we have to rely on the LWE assumption with large (but polynomial) approximation factors to prove the security.

New VRF Schemes. Following [HJ16], we say that a VRF scheme has "all the desired properties" if it has exponential-sized input space and a proof of adaptive security under a non-interactive complexity assumption. Here, we compare our schemes proposed in this paper with previous schemes that satisfy all the desired properties.

- In Sec. 6.1, we proposed new VRF scheme based on $\mathsf{F}_{\mathsf{MAH}}$. The scheme is parametrized by the parameters ℓ_1 and ℓ_2. By setting $\ell_1 = \ell$ and $\ell_2 = 1$, we obtain a new VRF scheme with very short proofs. They only consist of $\omega(\log \lambda)$ group elements.

- In Sec. 6.3, we proposed a variant of the above scheme. The verification keys consist of $\omega(\log \lambda)$ group elements and proofs consist of $\omega(\sqrt{\lambda} \log \lambda)$ group elements.

- In Appendix C, we proposed a new VRF scheme based on $\mathsf{F}_{\mathsf{AFF}}$. The verification key of the scheme only consists of $\omega(\log \lambda)$ group elements. However, the proof size of the scheme is large.

See Table 2 in Sec. 1 for the overview. From the table, we can see that all previous VRF schemes that satisfy all the desired properties [ACF14, BMR10, HW10, Jag15, HJ16] require $O(\lambda)$ group elements for *both* of verification keys and proofs. Our first scheme above significantly improves the size of proofs. Our second scheme improves both of the sizes of the verification keys and proofs. Compared to our second scheme, only advantage of our third scheme is that the reduction cost is better. Still, we think that our third scheme is also of interest because the construction is quite different from previous schemes.

Acknowledgement. The author is grateful to the members of the study group "Shin-Akarui-Angou-Benkyokai" and anonymous reviewers of Eurocrypt 2017 and Crypto 2017 for insightful comments. In particular, the author would like to thank Shuichi Katsumata for precious comments on the presentation and for pointing out that our initial construction can be slightly simplified. The author also specially thanks Goichiro Hanaoka for helpful comments. This work was supported by JSPS KAKENHI Grant Number 16K16068.

References

[ACF09] M. Abdalla, D. Catalano, and D. Fiore. Verifiable Random Functions from Identity-Based Key Encapsulation. In *EUROCRYPT*, pp. 554-571. 2009.

[ACF14] M. Abdalla, D. Catalano, and D. Fiore. Verifiable random functions: relations to identity-based key encapsulation and new constructions. In *J. Cryptology 27(3)*, pp. 544-593. 2014.

[ABB10a] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, pp. 553–572. 2010.

[ABB10b] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *CRYPTO*, pp. 98–115. 2010.

[Alp15] J. Alperin-Sheriff. Short signatures from homomorphic trapdoor functions. In *PKC*, pp. 236–255. 2015.

[AFL16] D. Apon, X. Fan, and F. Liu. Compact identity based encryption from LWE. In *IACR Cryptology ePrint Archive.* 2016:125, 2016.

[BGJS17] S. Badrinarayanan, V. Goyal, A. Jain, A. Sahai. A note on VRFs from verifiable functional encryption. In *IACR Cryptology ePrint Archive.* 2017:051, 2017.

[Bar89] David. A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *J. Comput. Syst. Sci.* 38(1), pp. 150–164. 1989.

[BCH86] P. Beame, S. A. Cook, and H. J. Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.* 15(4), pp. 994–1003. 1986.

[BHR12] M. Bellare, V. T. Hoang, and P. Rogaway. Foundations of garbled circuits. In *ACM-CCS*, pp. 784-796.

[BR09] M. Bellare and T. Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters IBE scheme. In *EUROCRYPT*, pp. 407–424. 2009.

[Bit17] N. Bitansky. Verifiable random functions from non-interactive witness-indistinguishable proofs. In *IACR Cryptology ePrint Archive.* 2017:18, 2017.

[BB04a] D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT*, pp. 223–238. 2004.

[BB04b] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *CRYPTO*, pp. 443–459. 2004.

[BHJ+13] F. Böhl, D. Hofheinz, T. Jager, J. Koch, J. Seo, and C. Striecks. Practical Signatures from Standard Assumptions. In *EUROCRYPT*, pp. 461–485, 2013.

[BF01] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pp. 213–229. 2001.

[BGG$^+$14] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, pp. 533–556. 2014.

[BGH07] D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryption without pairings. In *FOCS*, pp. 647–657. 2007.

[BMR10] D. Boneh, H. W. Montgomery, A. Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In *ACM-CCS*, pp.131–140. 2010.

[Boy10] X. Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *PKC*, pp. 499–517, 2010.

[BL16] X. Boyen and Qinyi Li. Towards tightly secure lattice short signature and ID-based encryption. In *ASIACRYPT*, to appear, 2016.

[BLP$^+$13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, pp. 575–584, 2013.

[BV16] Z. Brakerski and V. Vaikuntanathan. Circuit-ABE from LWE: Unbounded attributes and semi-adaptive security. In *CRYPTO (3)*, pp. 363–384, 2016.

[CHKP10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pp. 523–552, 2010.

[CLL+12] J. Chen, H. W. Lim, S. Ling, H. Wang, and H. Wee. Shorter IBE and Signatures via Asymmetric Pairings. In *Pairing*, pp. 122–140. 2012.

[Coc01] C. Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding*, pp. 360–363, 2001.

[Dod03] Efficient construction of (distributed) verifiable random functions. In *PKC*, pp. 1–17, 2003.

[DY05] Y. Dodis and A. Yampolskiy. A verifiable random function with short proofs and keys. In *PKC*, pp. 416–431, 2005.

[FHPS13] E. S. V. Freire, D. Hofheinz, K. G. Paterson, C. Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO (1)*, pp. 513–530. 2013.

[Gen06] C. Gentry. Practical identity-based encryption without random oracles. In *EUROCRYPT*, pp. 445–464. 2006.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pp. 197–206. 2008.

[Gol08] O. Goldreich. *Computational Complexity: A Conceptual Perspective.* Cambridge University Press, New York, NY, USA, 1 edition, 2008.

[GV15] S. Gorbunov and D. Vinayagamurthy. Riding on Asymmetry: Efficient ABE for Branching Programs. In *ASIACRYPT*, pp. 550–574, 2015.

[GKV10] S. D. Gordon, J. Katz, V. Vaikuntanathan. A Group Signature Scheme from Lattice Assumptions. In *ASIACRYPT*, pp. 395–412, 2010.

[GHKW17] R. Goyal, S. Hohenberger, V. Koppula, and B. Waters. A generic approach to constructing and proving verifiable random functions. In *IACR Cryptology ePrint Archive*. 2017:021, 2017.

[HJ16] D. Hofheinz and T. Jager. Verifiable random functions from standard assumptions. In *TCC (A1)*, pp. 336–362, 2016.

[HK08] D. Hofheinz and E. Kiltz. Programmable hash functions and their applications. In *CRYPTO*, pp. 21–38, 2008.

[HW10] S. Hohenberger and B. Waters. Constructing Verifiable Random Functions with Large Input Spaces. In *EUROCRYPT*, pp. 656–672. 2010.

[Jag15] T. Jager. Verifiable random functions from weaker assumptions. In *TCC(2)*, pp. 121–143, 2015.

[JR13] C. S. Jutla, A. Roy: Shorter Quasi-Adaptive NIZK Proofs for Linear Subspaces. In *ASIACRYPT (1)* pp. 1–20, 2013.

[KY16] S. Katsumata and S. Yamada. Partitioning via non-linear polynomial functions: More Compact IBEs from ideal lattices and bilinear maps. In *ASIACRYPT*, 2016, to appear.

[LOS$^+$10] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pp. 62–91. 2010.

[LW10] A. Lewko and B. Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In *TCC*, pp. 455–479. 2010.

[Lys02] A. Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. *CRYPTO*, pp. 597–612, 2002.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. *EUROCRYPT*, pp. 1–23, 2010.

[MRV99] S. Micali, M. O. Rabin, and S. P. Vadhan. Verifiable random functions. In *FOCS*, pp. 191–201, 1999.

[MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pp. 700–718, 2012.

[Nac07] D. Naccache. Secure and *practical* identity-based encryption. In *IET Information Security*, volume 1(2): pp.. 59–64, 2007.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC*, pp. 333–342. 2009.

[PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pp. 554-571, 2008.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pp. 84–93. ACM Press, 2005.

[SOK00] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairings. In *SCIS*, 2000. (In Japanese).

[Sha85] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pp. 47–53. 1985.

[SS96] M. Sipser and D. A. Spielman. Expander codes. In *IEEE Transactions on Information Theory*, 42(6):17101722, 1996.

[Wat05] B. Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pp. 114–127. 2005.

[Wat09] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO*, pp. 619–636. 2009.

[Yam16] S. Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In *EUROCRYPT (2)*, pp. 32–62. 2016.

[Zém01] G. Zémor. On expander codes. In *IEEE Transactions on Information Theory*, 47(2), pp. 835-837, 2001.

[ZCZ16] J. Zhang, Y. Chen, and Z. Zhang. Programmable hash functions from lattices: Short signatures and IBEs with small key sizes. In *CRYPTO*, pp. 303–332. 2016.

A Proof of Lemma 8

Proof. For $\{\mathbf{B}_i\}_{i\in[d]}$, we define $\mathbf{B}_j^\times$ for $j \in [d]$ recursively as follows:

$$\mathbf{B}_j^\times = \begin{cases} \mathbf{B}_1 & \text{if } j = 1 \\ \mathbf{B}_j \cdot \mathbf{G}^{-1}\left(\mathbf{B}_{j-1}^\times\right) & \text{if } j \geq 2. \end{cases}$$

The output of $\mathsf{PubMult}_d\left(\{\mathbf{B}_i\}_{i\in[d]}\right)$ is $\mathbf{B}_d^\times$. For $\mathbf{A}$ and $\{\mathbf{R}_i, x_i\}_{i\in[d]}$, we define $\mathbf{R}_j^\times$ for $j \in [d]$ recursively as follows:

$$\mathbf{R}_j^\times = \begin{cases} \mathbf{R}_1 & \text{if } j = 1 \\ \mathbf{R}_j \cdot \mathbf{G}^{-1}\left(\mathbf{A}\cdot\mathbf{R}_{j-1}^\times + \left(\prod_{i=1}^{j-1} x_i\right)\mathbf{G}\right) + x_j \cdot \mathbf{R}_{j-1}^\times & \text{if } j \geq 2. \end{cases}$$

The output of $\mathsf{TrapMult}_d\left(\mathbf{A}, \{\mathbf{R}_i, x_i\}_{i\in[d]}\right)$ is $\mathbf{R}_d^\times$. It is clear that $\mathsf{PubMult}$ and $\mathsf{TrapMult}$ run in polynomial time. We first show Eq.(12). The proof is by induction on d. The base case $d = 1$ is trivial. For $d \geq 2$, Let $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + x_i\mathbf{G}$ for $i \in [d]$. We have

$$\begin{aligned} \mathbf{B}_d^\times &= \mathbf{B}_d \cdot \mathbf{G}^{-1}(\mathbf{B}_{d-1}^\times) \\ &= (\mathbf{A}\mathbf{R}_d + x_d\mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{B}_{d-1}^\times) \\ &= \mathbf{A}\cdot\mathbf{R}_d\cdot\mathbf{G}^{-1}(\mathbf{B}_{d-1}^\times) + x_d \cdot \mathbf{B}_{d-1}^\times \\ &= \mathbf{A}\cdot\mathbf{R}_d\cdot\mathbf{G}^{-1}\left(\mathbf{A}\cdot\mathbf{R}_{d-1}^\times + \left(\prod_{i=1}^{d-1} x_i\right)\mathbf{G}\right) + x_d \cdot \left(\mathbf{A}\cdot\mathbf{R}_{d-1}^\times + \left(\prod_{i=1}^{d-1} x_i\right)\mathbf{G}\right) \qquad (33) \end{aligned}$$

$$\begin{aligned} &= \mathbf{A}\left(\mathbf{R}_d \cdot \mathbf{G}^{-1}\left(\mathbf{A}\cdot\mathbf{R}^{\times}_{d-1} + \left(\prod_{i=1}^{d-1} x_i\right)\mathbf{G}\right) + x_d\cdot \mathbf{R}^{\times}_{d-1}\right) + \left(\prod_{i=1}^{d} x_i\right)\mathbf{G} \\ &= \mathbf{A}\mathbf{R}^{\times}_d + \left(\prod_{i=1}^{d} x_i\right)\mathbf{G} \end{aligned}$$

where Eq.(33) follows from the induction hypothesis. Next, we proceed to show the bound on $\|\mathbf{R}^{\times}_d\|_\infty$. The proof is again by induction. The base case is trivial. For $d \geq 2$, we have

$$\begin{aligned} \|\mathbf{R}^{\times}_d\|_\infty &\leq \left\|\mathbf{R}_d \cdot \mathbf{G}^{-1}\left(\mathbf{A}\cdot\mathbf{R}^{\times}_{d-1} + \left(\prod_{i=1}^{d-1} x_i\right)\mathbf{G}\right)\right\|_\infty + |x_d|\cdot\left\|\mathbf{R}^{\times}_{d-1}\right\|_\infty \\ &\leq m\cdot\|\mathbf{R}_d\|_\infty + m(d-1)\delta \\ &= md\delta \end{aligned}$$

where the second inequality follows from $\mathbf{G}^{-1}\left(\mathbf{A}\cdot\mathbf{R}^{\times}_{d-1} + \left(\prod_{i=1}^{d-1} x_i\right)\mathbf{G}\right) \in \{0,1\}^{m\times m}$ and the induction hypothesis. This completes the proof of Lemma 8. □

B Proof of Lemma 11

Here, we prove Lemma 11. To prepare for the proof, we recall the following results regarding branching programs in previous works. Since the definition of branching programs is not necessary, we omit it and refer to [GV15].

Theorem 5 (Barrington's Theorem [Bar89])**.** *Every Boolean NAND circuit f that acts on ℓ inputs and has depth d can be computed by a width-5 permutation branching program Π of length 4^d. Given the description of the circuit f, the description of the branching program can be computed in* $\mathsf{poly}(\ell, 4^d)$ *time.*

The following lemma can be obtained by combining Lemma 6 and 7 of [GV15].

Lemma 15 (Adapted from [GV15])**.** *Let* $\mathsf{BP} : \{0,1\}^\ell \to \{0,1\}$ *be a width-5 permutation branching program of length L. There exist two efficient deterministic algorithms* BPPubEval *and* BPTrapEval *with the following properties.*

- $\mathsf{BPPubEval}\left(\mathsf{BP}, \{\mathbf{A}_i\}_{i\in[\ell]}, \{\mathbf{V}_{0,i}\}_{i\in[5]}, \mathbf{A}^{\mathsf{c}}\right) \to \mathbf{V}_{\mathsf{BP}} \in \mathbb{Z}_q^{n\times m}$. *Here,* $\mathbf{A}_i, \mathbf{V}_{0,i}, \mathbf{A}^{\mathsf{c}} \in \mathbb{Z}_q^{n\times m}$.
- $\mathsf{BPTrapEval}\left(\mathsf{BP}, \mathbf{A}, \{\mathbf{R}_i, x_i\}_{i\in[\ell]}, \{\mathbf{R}_{0,i}\}_{i\in[5]}, \mathbf{R}^{\mathsf{c}}\right) \to \mathbf{R}_{\mathsf{BP}} \in \mathbb{Z}^{m\times m}$. *Here,* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{R}_i, \mathbf{R}_{0,i}, \mathbf{R}^{\mathsf{c}} \in \{-1,0,1\}^{m\times m}$, $x_i \in \{0,1\}$. *Furthermore, we have*

$$\begin{aligned} \mathsf{BPPubEval}\left(\mathsf{BP}, \{\mathbf{A}\mathbf{R}_i - x_i\mathbf{G}\}_{i\in[\ell]}, \{\mathbf{A}\mathbf{R}_{0,i} - y_i\mathbf{G}\}_{i\in[5]}, \mathbf{A}\mathbf{R}^{\mathsf{c}} - \mathbf{G}\right) &= \mathbf{A}\mathbf{R}_{\mathsf{BP}} - \mathsf{BP}(x)\mathbf{G} \\ \textit{and} \qquad \|\mathbf{R}_{\mathsf{BP}}\|_\infty &\leq (3m+1)L, \end{aligned}$$

  where $x \in \{0,1\}^\ell$ is the concatenation of $x_1, \ldots, x_\ell$, $y_1 = 1$, and $y_j = 0$ for $j \in [2,5]$.

We note that Lemma 15 is only shown for the case of $\mathbf{R}_i, \mathbf{R}_{0,i} \in \{-1,1\}^{m\times m}$ in [GV15]. However, the same proof works without change for the case of $\mathbf{R}_i, \mathbf{R}_{0,i} \in \{-1,0,1\}^{m\times m}$ as well. We then proceed to prove Lemma 11.

Proof of Lemma 11. We give the description of $\mathsf{NC}^1\mathsf{PubEval}$ and $\mathsf{NC}^1\mathsf{TrapEval}$.

$\mathsf{NC^1PubEval}\left(\mathsf{F}, \{\mathbf{B}_i\}_{i\in[\ell]}\right)$: It first expresses $\mathsf{F} : \{0,1\}^\ell \to \{0,1\}$ as a Boolean NAND circuit of depth d. Then, it converts it into a width-5 branching BP of length 4^d using Theorem 5. It then sets $\mathbf{A}_i = -\mathbf{B}_i$ for $i \in [\ell]$, $\mathbf{A}^{\mathsf{c}} = -\mathbf{G}$, $\mathbf{V}_{0,1} := -\mathbf{G}$, and $\mathbf{V}_{0,j} := \mathbf{0}_{n\times m}$ for $j \in [2,5]$. Finally, it runs $\mathsf{BPPubEval}$ in Lemma 15 as $\mathsf{BPPubEval}\left(\mathsf{BP}, \{\mathbf{A}_i\}_{i\in[\ell]}, \{\mathbf{V}_{0,i}\}_{i\in[5]}, \mathbf{A}^{\mathsf{c}}\right) \to \mathbf{V}_{\mathsf{BP}} \in \mathbb{Z}_q^{n\times m}$ and outputs $\mathbf{B}_{\mathsf{F}} := -\mathbf{V}_{\mathsf{BP}}$.

$\mathsf{NC^1TrapEval}\left(\mathsf{F}, \mathbf{A}, \{\mathbf{R}_i, x_i\}_{i\in[\ell]}\right)$: It first converts F into a branching program BP. It then sets $\mathbf{R}'_i := -\mathbf{R}_i$ for $i \in [\ell]$ and $\mathbf{R}_{0,j} = \mathbf{R}^{\mathsf{c}} = \mathbf{0}_{m\times m}$ for all $j \in [5]$. Finally, it runs $\mathsf{BPTrapEval}(\mathsf{BP}, \mathbf{A}, \{\mathbf{R}'_i, x_i\}_{i\in[\ell]}, \{\mathbf{R}_{0,i}\}_{i\in[5]}, \mathbf{R}^{\mathsf{c}}) \to \mathbf{R}_{\mathsf{BP}} \in \mathbb{Z}^{m\times m}$ and outputs $\mathbf{R}_{\mathsf{F}} := -\mathbf{R}_{\mathsf{BP}}$.

We then prove that these algorithms satisfy the required properties. It can be seen by Theorem 5 that the requirement on the running time is satisfied. From Lemma 15, it follows that $\|\mathbf{R}_{\mathsf{F}}\|_\infty = \|\mathbf{R}_{\mathsf{BP}}\|_\infty \le (3m+1)4^d$ since the length of BP is 4^d. It remains to show that $\mathbf{B}_{\mathsf{F}} = \mathbf{A}\mathbf{R}_{\mathsf{F}} + \mathsf{F}(x)\mathbf{G}$ holds if $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + x_i\mathbf{G}$ for $i \in [\ell]$. We have $\mathbf{A}_i = \mathbf{A}\mathbf{R}'_i - x_i\mathbf{G}$ for $i \in [\ell]$, $\mathbf{V}_{0,1} = \mathbf{A}\mathbf{R}_{0,1} - \mathbf{G}$, $\mathbf{V}_{0,j} = \mathbf{A}\mathbf{R}_{0,j}$ for $j \in [2,5]$, and $\mathbf{A}^{\mathsf{c}} = \mathbf{A}\mathbf{R}^{\mathsf{c}} - \mathbf{G}$. Therefore, $\mathbf{V}_{\mathsf{BP}} = \mathbf{A}\mathbf{R}_{\mathsf{BP}} - \mathsf{BP}(x)\cdot\mathbf{G}$ follows from Lemma 15. Then, $\mathbf{B}_{\mathsf{F}} = \mathbf{A}\mathbf{R}_{\mathsf{F}} + \mathsf{F}(x)\cdot\mathbf{G}$ immediately follows from $\mathsf{F}(x) = \mathsf{BP}(x)$. This completes the proof of Lemma 11. □

C VRF Scheme Based on $\mathsf{F}_{\mathsf{AFF}}$

Here, we construct a VRF scheme based on the partitioning function $\mathsf{F}_{\mathsf{AFF}}$. We first introduce some notations regarding circuits following [BHR12]. We use simpler definition adapted to our setting.

Circuit. We define a circuit with a single output wire as a 4-tuple $f = (u, v, \mathsf{A}, \mathsf{B})$. For simplicity, we consider circuits with only NAND gates. Here, u is the number of inputs and v is the number of gates. We let $u+v$ be the number of wires. We let $\mathrm{Inputs} = [u]$, $\mathrm{Wires} = [u+v]$, $\mathrm{OutputWire} = \{u+v\}$, and $\mathrm{Gates} = [u+1, \ldots, u+v]$. Then $\mathsf{A} : \mathrm{Gates} \to \mathrm{Wires}\backslash\mathrm{OutputWire}$ is a function to identify each gate's first incoming wire and $\mathsf{B} : \mathrm{Gates} \to \mathrm{Wires}\backslash\mathrm{OutputWire}$ is a function that identify each gate's second incoming wire. We require $\mathsf{A}(i) < \mathsf{B}(i) < i$ for all $i \in \mathrm{Gates}$. The depth of a gate is defined as the maximum distance from the input gate to the gate. We define the depth of the circuit as the depth of the output gate.

Evaluating an Circuit. To evaluate the circuit f on input $x = x_1x_2\cdots x_u \in \{0,1\}^u$, we first assign each $x_i \in \{0,1\}$ to the gate i for $i \in [u]$. Then, for $i \in [u+1, u+v]$, we compute $x_i = \neg(x_{\mathsf{A}(i)} \wedge x_{\mathsf{B}(i)})$. The final output is x_{u+v}. Similarly, we can evaluate the circuit on input $x = (x_1, \ldots, x_u) \in \mathbb{Z}_p^u$ for any integer p. To do this, we first assign each $x_i \in \mathbb{Z}_p$ to the gate i for $i \in [u]$. Then, for $i \in [u+1, u+v]$, we compute $x_i = 1 - x_{\mathsf{A}(i)}x_{\mathsf{B}(i)} \in \mathbb{Z}_p$. The final output is x_{u+v}.

Construction. We let the input space of the scheme be $\mathcal{X} = \{0,1\}^k$ and the output space be $\mathcal{Y} = \mathbb{G}_T$. Let us use the same notation as in Sec. 4.4. We also define $u := 3\eta + k$. Let $\neg\mathsf{F}_{\mathsf{AFF}} : \{0,1\}^u \to \{0,1\}$ be a function defined as $\neg\mathsf{F}_{\mathsf{AFF}}(x) := 1 - \mathsf{F}_{\mathsf{AFF}}(x)$. By Lemma 10, $\neg\mathsf{F}_{\mathsf{AFF}} : \{0,1\}^u \to \{0,1\}$ can be computed by an (efficiently computable) circuit $f_{\mathsf{AFF}} = (u, v, \mathsf{A}, \mathsf{B})$ such that $v = \mathsf{poly}(u)$ and depth $O(\log u)$. This circuit will be used in the construction.

$\mathsf{Gen}(1^\lambda)$: On input 1^λ, it chooses group description $\Pi \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$. It chooses random generators $g, h \xleftarrow{\$} \mathbb{G}^*$, $w_0 \xleftarrow{\$} \mathbb{Z}_p^*$, and $w_1, \ldots, w_{u-k} \xleftarrow{\$} \mathbb{Z}_p$. It then outputs

$$\mathsf{vk} = \Big(\ \Pi,\ g,\ h,\ \{W_i := g^{w_i}\}_{i\in[0,u-k]}\ \Big) \qquad \text{and} \qquad \mathsf{sk} = \big(\ \{w_i\}_{i\in[0,u-k]}\ \big).$$

$\mathsf{Eval}(\mathsf{sk}, X)$: Given $X \in \{0,1\}^k$, it first computes $f_{\mathsf{AFF}} = (u, v, \mathsf{A}, \mathsf{B})$. Then, it sets

$$\theta_i = \begin{cases} w_i & \text{if } i \in [u-k] \\ X_{i-(u-k)} & \text{if } i \in [u-k+1, u] \\ 1 - \theta_{\mathsf{A}(i)} \cdot \theta_{\mathsf{B}(i)} & \text{if } i \in [u+1, u+v] \end{cases},$$

where X_j is the j-th bit of $X \in \{0,1\}^k$. Note that $\theta_i \in \mathbb{Z}_p$ is the value associated to the i-th wire when evaluating the circuit on input $(w_1, w_2, \cdots, w_{u-k}, X_1, \ldots, X_k) \in \mathbb{Z}_p^u$. The output of the circuit is $\theta := \theta_{u+v}$. It then computes and outputs

$$Y = e(g,h)^{\theta/w_0} \qquad \text{and} \qquad \pi = \left(\pi_0 = g^{\theta/w_0}, \ \left\{ \pi_i = g^{\theta_i} \right\}_{i \in [u+1, u+v]} \right).$$

$\mathsf{Verify}(\mathsf{vk}, X, Y, \pi)$: It first checks the validity of vk by the following steps. It outputs 0 if any of the following does not hold:

1. vk is in the form of $(\Pi, g, h, \{W_i\}_{i \in [0, u-k]})$.
2. $\mathsf{GrpVfy}(\Pi) \to 1$, $g, h, W_0 \in \mathbb{G}^*$, and $W_i \in \mathbb{G}$ for all $i \in [u-k]$.

It then checks the validity of Y and π. To do so, it sets $\pi_i := W_i$ for $i \in [u-k]$ and $\pi_i := g^{X_{i-(u-k)}}$ for $i \in [u-k+1, u]$. Furthermore, it computes the description of $f_{\mathsf{AFF}} = (u, v, \mathsf{A}, \mathsf{B})$. It outputs 0 if any of the following does not hold:

3. $X \in \{0,1\}^k$, $Y \in \mathbb{G}_T$, π is in the form of $\pi = (\pi_0 \in \mathbb{G}, \{\pi_i \in \mathbb{G}\}_{i \in [u+1, u+v]})$.
4. For all $i \in [u+1, u+v]$, $e(g, \pi_i) = e(g, g) \cdot \left(e\left(\pi_{\mathsf{A}(i)}, \pi_{\mathsf{B}(i)}\right)\right)^{-1}$ holds.
5. $e(\pi_0, h) = Y$ and $e(\pi_0, W_0) = e(\pi_{u+v}, g)$ hold.

If all the above conditions hold, it outputs 1.

Correctness. We prove correctness of the scheme. Let us assume that (vk, Y, π) are correctly generated as $(\mathsf{vk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ and $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$. It is straightforward to see that it passes Step 1, 2, and 3 in the verification algorithm. It also passes Step 4, because $\theta_i = 1 - \theta_{\mathsf{A}(i)} \cdot \theta_{\mathsf{B}(i)}$ holds for all wire $i \in [u+1, u+v]$. Finally, by a simple calculation, it can be seen that it also passes Step 5. This completes the proof of the correctness.

Unique Provability. We prove the unique provability of the scheme. We have to show that for any $\mathsf{vk} \in \{0,1\}^*$, there does not exist any (Y, π, Y', π') such that $\mathsf{Verify}(\mathsf{vk}, X, Y, \pi) = \mathsf{Verify}(\mathsf{vk}, X, Y', \pi') = 1$ and $Y \neq Y'$.

- Step 1 checks whether Π contains valid certified bilinear group parameters. Step 2 checks whether $g, h, W_0 \in \mathbb{G}^*$ and $W_i \in \mathbb{G}$. Thus, we may assume in the following that all these group elements are valid and have a unique encoding. In particular, there are unique $w_i \in \mathbb{Z}_p$ such that $g^{w_i} = W_i$ for all $i \in [0, u-k]$.

- In Step 3, it is checked that whether $X \in \{0,1\}^k$, $Y \in \mathbb{G}_T$, and π is in the form of $\pi = (\{\pi_i \in \mathbb{G}\}_{i \in [u+1, u+v]})$. For $i \in [u+v]$, let us define θ_i as the unique value that is assigned for gate i when we evaluate the circuit f_{AFF} on input $(w_1, \ldots, w_{u-k}, X_1, \ldots, X_k)$. It can be seen by induction that in order to pass Step 4, $\log_g \pi_i = \theta_i$ should hold for all $i \in [u+v]$. Finally, Step 5 checks that $\pi_0 = g^{\theta_{u+v}/w_0}$ and $Y = e(g,h)^{\theta_{u+v}/w_0}$. Due to the fact that the bilinear group is certified, which guarantees that each group element has a unique encoding and that the bilinear map is non-degenerate, the unique provability follows.

This completes the proof of the unique provability.

Pseudorandomness. We have the following theorem.

Theorem 6. *The above scheme satisfies pseudorandomness assuming* L-DDH *with* $L = 2^d$*, where* d *is the depth of the circuit* f_{AFF}*. Since* f_{AFF} *is a* $O(\log \lambda)$*-depth circuit, the above scheme is secure assuming* L-DDH *with polynomially bounded* L*.*

Proof. The proof of the theorem roughly follows that of Theorem 4. We will highlight the difference. Let $\mathcal{A}$ be a PPT adversary that breaks pseudorandomness of the scheme. In addition, let $\epsilon = \epsilon(\lambda)$ and $Q = Q(\lambda)$ be its advantage and the upper bound on the number of the evaluation queries, respectively. Let $\epsilon_0(\lambda)$ be a noticeable function such that $\epsilon(\lambda) \geq \epsilon_0(\lambda)$ holds for infinitely many λ. We show the security of the scheme via the following sequence of games. In each game, a value $\mathsf{coin}' \in \{0,1\}$ is defined. In the following, we define E_i be the event that $\mathsf{coin}' = \mathsf{coin}$.

Game_0 : This is the real security game. At the end of the game, the challenger outputs the same bit as $\mathcal{A}$. Namely, $\mathsf{coin}' = \widehat{\mathsf{coin}}$. We have

$$\left|\Pr[\mathsf{E}_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon.$$

Game_1 : In this game, we change Game_0 so that the challenger performs the following additional step at the end of the game. First, the challenger runs $K \stackrel{\$}{\leftarrow} \mathsf{PrtSmp}_{\mathsf{AFF}}(1^\lambda, Q, \epsilon_0)$ and checks whether the following condition holds:

$$\mathsf{F}_{\mathsf{AFF}}(K, X^{(1)}) = \cdots = \mathsf{F}_{\mathsf{AFF}}(K, X^{(Q)}) = 1 \ \wedge \ \mathsf{F}_{\mathsf{AFF}}(K, X^\star) = 0$$

where $X^\star$ is chosen by $\mathcal{A}$ at the challenge phase, and $X^{(1)}, \dots, X^{(Q)}$ are inputs to the VRF for which $\mathcal{A}$ has queried the evaluation of the function. If it does not hold, the challenger ignores the output $\widehat{\mathsf{coin}}$ of $\mathcal{A}$, and sets $\mathsf{coin}' \stackrel{\$}{\leftarrow} \{0,1\}$. In this case, we say that the challenger aborts. If the above condition holds, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. Similarly to the proof of Theorem 4, we have that

$$\left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| \geq \tau$$

holds for infinitely many λ. Here, $\tau(\lambda)$ is a noticeable function that is specified by ϵ_0, Q, and the underlying partitioning function $\mathsf{F}_{\mathsf{AFF}}$ (See Theorem 2).

Game_2 : In this game, we change the way w_i are chosen. At the beginning of the game, the challenger picks $K \stackrel{\$}{\leftarrow} \mathsf{PrtSmp}_{\mathsf{AFF}}(1^\lambda, Q, \epsilon_0)$. We assume $K \in \{0,1\}^{u-k}$, which holds for all sufficiently large λ. It then samples $\alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$, and $\tilde{w}_i \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ for $i \in [0, u-k]$. Then, w_i are defined as

$$w_0 := \tilde{w}_0 \cdot \alpha, \qquad w_i = \tilde{w}_i \cdot \alpha + K_i \qquad \text{for } i \in [u-k]$$

where K_i is the i-th bit of K. The rest of the game is the same as in Game_1. The statistical distance of the distributions of $\{w_i\}_{i \in [u-k]}$ in Game_1 and Game_2 is at most $(u-k)/p$, which is negligible. Therefore, we have

$$|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(\lambda).$$

Before describing the next game, we introduce additional notations. For $\{\tilde{w}_i\}_{i\in[u-k]}$, $K \in \{0,1\}^{u-k}$, and $X \in \{0,1\}^k$, let us define polynomials $\{\ \mathsf{P}_{X,i}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]\ \}_{i\in[u+v]}$ as follows.

$$\mathsf{P}_{X,i}(\mathsf{Z}) = \begin{cases} \tilde{w}_i \cdot \mathsf{Z} + K_i & \text{if } i \in [u-k] \\ X_{i-(u-k)} & \text{if } i \in [u-k+1, u] \\ 1 - \mathsf{P}_{X,\mathsf{A}(i)}(\mathsf{Z}) \cdot \mathsf{P}_{X,\mathsf{B}(i)}(\mathsf{Z}) & \text{if } i \in [u+1, u+v] \end{cases}$$

Furthermore, we denote $\mathsf{P}_X(\mathsf{Z}) := \mathsf{P}_{X,u+v}(\mathsf{Z})$. It can be seen that $\mathsf{P}_{X,i}(\alpha) = \theta_i$ and $\mathsf{P}_X(\alpha) = \theta$. We state the following lemma, which plays an important roll in our security proof.

Lemma 16. *There exists* $\mathsf{R}_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ *such that*

$$\mathsf{P}_X(\mathsf{Z}) = \begin{cases} 1 + \mathsf{Z} \cdot \mathsf{R}_X(\mathsf{Z}) & \text{if} \quad \mathsf{F}_{\mathsf{AFF}}(K,X) = 0 \\ \mathsf{Z} \cdot \mathsf{R}_X(\mathsf{Z}) & \text{if} \quad \mathsf{F}_{\mathsf{AFF}}(K,X) = 1 \end{cases}.$$

In other words, the constant term of $\mathsf{P}_X(\mathsf{Z})$ *is* $1 - \mathsf{F}_{\mathsf{AFF}}(K,X)$*. We have* $\deg(\mathsf{P}_X(\mathsf{Z})), \deg(\mathsf{P}_{X,i}(\mathsf{Z})) \le 2^d$ *for all* $i \in [u+v]$*, where* d *is the depth of the circuit* f_{AFF}*.*

So as not to interrupt the proof of Theorem 4, we intentionally skip the proof of Lemma 16 for the time being.

Game_3 In this game, we change the game so that the challenger aborts as soon as the abort condition becomes true. Since this is only a conceptual change, we have

$$\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3].$$

Game_4 In this game, we change the way the evaluation queries are answered. When $\mathcal{A}$ makes a query for an input X, the challenger first checks whether $\mathsf{F}_{\mathsf{AFF}}(K,X) = 0$ and aborts if so (as specified in Game_3). Otherwise, it computes $\mathsf{R}_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{P}_X(\mathsf{Z}) = \mathsf{Z} \cdot \mathsf{R}_X(\mathsf{Z})$ and returns

$$Y = e\left(g^{\mathsf{R}_X(\alpha)/\tilde{w}_0}, h\right), \quad \pi = \left(\ \pi_0 = g^{\mathsf{R}_X(\alpha)/\tilde{w}_0},\ \left\{\ \pi_i = g^{\mathsf{P}_{X,i}(\alpha)}\ \right\}_{i\in[u+1,u+v]}\ \right) \tag{34}$$

to $\mathcal{A}$. Note that such $\mathsf{R}_X(\mathsf{Z})$ exists by Lemma 16. We claim that this is only a conceptual change. First, π_i for $i \in [u+1, u+v]$ are correctly generated because $\mathsf{P}_{X,i}(\alpha) = \theta_i$. Furthermore, we have

$$\frac{\mathsf{R}_X(\alpha)}{\tilde{w}_0} = \frac{\alpha \cdot \mathsf{R}_X(\alpha)}{\tilde{w}_0 \alpha} = \frac{\mathsf{P}_X(\alpha)}{w_0} = \frac{\theta}{w_0}.$$

Thus, π_0 and Y are also correctly generated. These indicate that the simulation by the challenger is perfect. Since the view of $\mathcal{A}$ is unchanged, we have

$$\Pr[\mathsf{E}_3] = \Pr[\mathsf{E}_4].$$

Game_5 : In this game, we change the way the challenge value $Y_0^\star = \mathsf{Eval}(\mathsf{sk}, X^\star)$ is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to generate $Y_0^\star$, it first computes $\mathsf{R}_{X^\star}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{P}_{X^\star}(\mathsf{Z}) = 1 + \mathsf{Z} \cdot \mathsf{R}_{X^\star}(\mathsf{Z})$. By Lemma 16, such $\mathsf{R}_{X^\star}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ exists. It then sets

$$Y_0^\star = \left(e\left(g,h\right)^{1/\alpha} \cdot e\left(g^{\mathsf{R}_{X^\star}(\alpha)}, h\right)\right)^{1/\tilde{w}_0}$$

and returns it to $\mathcal{A}$. We claim that this is only a conceptual change. This can be seen by observing

$$\left(e\left(g,h\right)^{1/\alpha}\cdot e\left(g^{\mathsf{R}_{X^\star}(\alpha)},h\right)\right)^{1/\tilde{w}_0}=e\left(g^{(1+\alpha\mathsf{R}_{X^\star}(\alpha))/\tilde{w}_0\alpha},h\right)=e\left(g^{\mathsf{P}_{X^\star}(\alpha)/w_0},h\right)=e\left(g^{\theta/w_0},h\right).$$

Since the view of $\mathcal{A}$ is unchanged, we conclude that

$$\Pr[\mathsf{E}_4]=\Pr[\mathsf{E}_5].$$

Game_6 In this game, we change the challenge value to a random element in $\mathbb{G}_T$ regardless of whether $\mathsf{coin}=0$ or $\mathsf{coin}=1$. Namely, Game_6 challenger sets $Y_0^\star\overset{\$}{\leftarrow}\mathbb{G}_T$.

We claim that $|\Pr[\mathsf{E}_5]-\Pr[\mathsf{E}_6]|$ is negligible assuming L-DDH with $L=2^d$. To show this, we construct an adversary $\mathcal{B}$ against the problem using $\mathcal{A}$, which is described as follows.

$\mathcal{B}$ is given the problem instance $(\Pi,g,h,\{g^{\alpha^i}\}_{i\in[L]},\Psi)$ of L-DDH where $\Psi=e(g,h)^{1/\alpha}$ or $\Psi\overset{\$}{\leftarrow}\mathbb{G}_T$. At any point in the game, $\mathcal{B}$ aborts and sets $\mathsf{coin}'\overset{\$}{\leftarrow}\{0,1\}$ if the abort condition is satisfied. It first computes

$$W_0=(g^\alpha)^{\tilde{w}_0}\qquad\text{and}\qquad W_i=(g^\alpha)^{\tilde{w}_i}\cdot g^{K_i}\quad\text{for}\quad i\in[u-k]$$

and returns $\mathsf{vk}=(\Pi,g,h,\{W_i\}_{i\in[0,u-k]})$ to $\mathcal{A}$. When $\mathcal{A}$ makes an evaluation query on input X, it computes (Y,π) as Eq.(34) and returns it to $\mathcal{A}$. These terms can be efficiently computable from the problem instance without knowing α, because $\mathsf{R}_X(\alpha)$ and $\mathsf{P}_{X,i}(\alpha)$ are polynomials in α with degree at most $L=2^d$ by Lemma 16. When $\mathcal{A}$ makes the challenge query on input $X^\star$, $\mathcal{B}$ first picks $\mathsf{coin}\overset{\$}{\leftarrow}\{0,1\}$ and returns $Y^\star\overset{\$}{\leftarrow}\mathbb{G}$ if $\mathsf{coin}=1$. Otherwise, it returns

$$Y^\star=\left(\Psi\cdot e\left(g^{\mathsf{R}_{X^\star}(\alpha)},h\right)\right)^{1/\tilde{w}_0}$$

to $\mathcal{A}$. Note that $g^{\mathsf{R}_{X^\star}(\alpha)}$ can be efficiently computed from the problem instance because the degree of $\mathsf{R}_{X^\star}(\alpha)$ is at most L. At the end of the game, coin' is defined. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}'=\mathsf{coin}$ and 0 otherwise.

It can easily be seen that the view of $\mathcal{A}$ corresponds to that of Game_5 if $\Psi=e(g,h)^{1/\alpha}$ and Game_6 if $\Psi\overset{\$}{\leftarrow}\mathbb{G}_T$. It is clear that the advantage of $\mathcal{B}$ is $|\Pr[\mathsf{E}_5]-\Pr[\mathsf{E}_6]|$. Assuming L-DDH, we have

$$|\Pr[\mathsf{E}_5]-\Pr[\mathsf{E}_6]|=\mathsf{negl}(\lambda).$$

Analysis. From the above, we have

$$\begin{aligned}\left|\Pr[\mathsf{E}_6]-\frac{1}{2}\right| &= \left|\Pr[\mathsf{E}_1]-\frac{1}{2}+\sum_{i=1}^{5}\Pr[\mathsf{E}_{i+1}]-\Pr[\mathsf{E}_i]\right|\\ &\geq \left|\Pr[\mathsf{E}_1]-\frac{1}{2}\right|-\sum_{i=1}^{5}|\Pr[\mathsf{E}_{i+1}]-\Pr[\mathsf{E}_i]|\\ &\geq \tau(\lambda)-\mathsf{negl}(\lambda).\end{aligned}$$

for infinitely many λ. Since $\Pr[\mathsf{E}_6]=1/2$, this implies $\tau(\lambda)\leq\mathsf{negl}(\lambda)$ for infinitely many λ, which is a contradiction. □

To complete the proof of Theorem 6, it remains to prove Lemma 16.

Proof of Lemma 16. We prove the following (stronger) claim.

Claim 1. *For $i \in [u+v]$, let b_i be the value assigned to wire i when we evaluate the circuit f_{AFF} on input $(K, X) = (K_1, \dots, K_{u-k}, X_1, \dots, X_k) \in \{0,1\}^u$. Furthermore, let d_i be the depth of gate i. Then, there exists $\mathsf{R}_{X,i}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that*

$$\mathsf{P}_{X,i}(\mathsf{Z}) = b_i + \mathsf{Z} \cdot \mathsf{R}_{X,i}(\mathsf{Z}).$$

Furthermore, we have $\deg(\mathsf{P}_{X,i}(\mathsf{Z})) \le 2^{d_i}$.

Proof of Claim 1. The proof is by induction on i. It is clear that the claim holds for $i \in [u]$. Let us assume that the claim holds for all i in $[j]$ with $j \ge u$. Then, we have

$$\begin{aligned}
& \mathsf{P}_{X,j+1}(\mathsf{Z}) \\
= \; & 1 - \mathsf{P}_{X,\mathsf{A}(j+1)}(\mathsf{Z}) \cdot \mathsf{P}_{X,\mathsf{B}(j+1)}(\mathsf{Z}) \\
= \; & 1 - \big(b_{\mathsf{A}(j+1)} + \mathsf{Z} \cdot \mathsf{R}_{X,\mathsf{A}(j+1)}(\mathsf{Z})\big) \cdot \big(b_{\mathsf{B}(j+1)} + \mathsf{Z} \cdot \mathsf{R}_{X,\mathsf{B}(j+1)}(\mathsf{Z})\big) \qquad (35) \\
= \; & \underbrace{(1 - b_{\mathsf{A}(j+1)} \cdot b_{\mathsf{B}(j+1)})}_{=b_{j+1}} + \\
& \mathsf{Z} \cdot \underbrace{\big(-b_{\mathsf{A}(j+1)} \cdot \mathsf{R}_{X,\mathsf{B}(j+1)}(\mathsf{Z}) - b_{\mathsf{B}(j+1)} \cdot \mathsf{R}_{X,\mathsf{A}(j+1)}(\mathsf{Z}) + \mathsf{Z} \cdot \mathsf{R}_{X,\mathsf{A}(j+1)}(\mathsf{Z}) \cdot \mathsf{R}_{X,\mathsf{B}(j+1)}(\mathsf{Z})\big)}_{:=\mathsf{R}_{X,j+1}(\mathsf{Z})} \\
= \; & b_{j+1} + \mathsf{Z} \cdot \mathsf{R}_{X,j+1}(\mathsf{Z})
\end{aligned}$$

where we introduced two polynomials $\mathsf{R}_{X,\mathsf{A}(j+1)}(\mathsf{Z})$ and $\mathsf{R}_{X,\mathsf{B}(j+1)}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ in Eq.(35). The existence of these polynomials is implied by $\mathsf{A}(j+1), \mathsf{B}(j+1) < j+1$ and by the induction hypothesis. We also have

$$\deg(\mathsf{P}_{X,j+1}(\mathsf{Z})) = \deg\big(1 - \mathsf{P}_{X,\mathsf{A}(j+1)}(\mathsf{Z}) \cdot \mathsf{P}_{X,\mathsf{B}(j+1)}(\mathsf{Z})\big) \le 2^{d_{\mathsf{A}(j+1)}} + 2^{d_{\mathsf{B}(j+1)}} \le 2^{d_{j+1}},$$

where the first inequality follows from the induction hypothesis and the second from $d_{j+1} = \max\{d_{\mathsf{A}(j+1)}, d_{\mathsf{B}(j+1)}\} + 1$. This completes the proof of the claim. □

The proof of the lemma completes by observing $d = d_{u+v}$, $b_{u+v} = 1 - \mathsf{F}_{\mathsf{AFF}}(K, X)$, and $\mathsf{P}_X(\mathsf{Z}) = \mathsf{P}_{X,u+v}(\mathsf{Z})$. □

Contents