Efficient Adaptively-Secure IB-KEMs and VRFs via Near-Collision Resistance*

Tibor Jager[1], Rafael Kurek[1] and David Niehues[2]
[1]Bergische Universität Wuppertal, Wuppertal, Germany
tibor.jager@uni-wuppertal.de, rafael.kurek@rub.de
[2]Paderborn University, Paderborn, Germany, david.niehues@upb.de

February 13, 2021

Abstract

We construct more efficient cryptosystems with provable security against *adaptive* attacks, based on simple and natural hardness assumptions in the standard model. Concretely, we describe:

- An adaptively-secure variant of the efficient, selectively-secure LWE-based identity-based encryption (IBE) scheme of Agrawal, Boneh, and Boyen (EUROCRYPT 2010). In comparison to the previously most efficient such scheme by Yamada (CRYPTO 2017) we achieve smaller lattice parameters and shorter public keys of size $\mathcal{O}(\log \lambda)$, where λ is the security parameter.
- Adaptively-secure variants of two efficient selectively-secure pairing-based IBEs of Boneh and Boyen (EUROCRYPT 2004). One is based on the DBDH assumption, has the same ciphertext size as the corresponding BB04 scheme, and achieves full adaptive security with public parameters of size only $\mathcal{O}(\log \lambda)$. The other is based on a q-type assumption and has public key size $\mathcal{O}(\lambda)$, but a ciphertext is only a single group element and the security reduction is quadratically tighter than the corresponding scheme by Jager and Kurek (ASIACRYPT 2018).
- A very efficient adaptively-secure verifiable random function where proofs, public keys, and secret keys have size $\mathcal{O}(\log \lambda)$.

As a technical contribution we introduce *blockwise partitioning*, which leverages the assumption that a cryptographic hash function is *weak near-collision resistant* to prove full adaptive security of cryptosystems.

1 Introduction

A very fundamental question in cryptography is to which extent idealizations like the random oracle model [BR93] are necessary to obtain practical constructions of cryptosystems. By advancing our techniques to prove security of schemes, we may eventually be able to obtain standard-model schemes that are about as efficient as corresponding schemes with security proofs in the ROM. From a practical perspective, it would be preferable to have security guarantees that are not based on an uninstantiable model [CGH98].

*This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Centre On-The-Fly Computing (GZ: SFB 901/3) under the project number 160364472. Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement 802823.

From a theoretical perspective, it allows us to understand when a random oracle is necessary, and when not. For some primitives it is known that a programmable random oracle is indeed inherently necessary [Nie02, FLR+10, HMS12, FF13]. But for many others, including those considered in this paper, there are no such impossibility results.

In the context of identity-based encryption the established standard security notion [BF01] considers an adversary which is able to choose the identities for which it requests secret keys or a challenge ciphertext *adaptively* in the security experiment. This yields much stronger security guarantees than so-called *selective* security definitions [BB04a], where the adversary has to announce the "target identity" associated with a challenge ciphertext at the beginning of the security experiment, even before seeing the public parameters.

"Selective" security is much easier to achieve and therefore yields more efficient constructions. The random oracle model is then a useful tool to generically convert a selectively-secure scheme into an adaptively-secure one. This has negligible performance overhead, and thus yields an efficient and adaptively-secure construction. This generic construction is based on the fact that a random oracle is "programmable", which essentially means that it is possible to adaptively modify the mapping of function inputs to outputs in a way that is convenient for the security proof. While this is very useful to achieve efficient and adaptively-secure constructions, it is often considered a particularly unnatural property of the random oracle model, due to the fact that no fixed function can be as freely adaptively programmed as a random oracle.

There exist techniques to achieve adaptive security in the standard model by realizing certain properties of a random oracle with a concrete construction (i. e., in the standard model). This includes *admissible hash functions* [BB04b], *programmable hash functions* [Wat05, HK08, HJK11, FHPS13, CFN15], and *extremely lossy functions* [Zha16]. However, these typically yield significantly less efficient constructions and are therefore less interesting for practical applications than corresponding constructions in the random oracle model.

A recent, quite different approach that addresses this issue is to use *truncation collision resistance* [JK18] of a cryptographic hash function to achieve adaptive security. In contrast to the aforementioned approaches, this does not introduce a new "algebraic" construction of a hash function. Instead, their idea is to formulate a concrete hardness assumption that on the one hand is "weak enough" to appear reasonable for *standard* cryptographic hash functions, such as SHA-3, but which at the same time is "strong enough" to be used to achieve adaptive security. It is shown that this indeed yields very efficient and adaptively-secure constructions, such as identity-based encryption with a single group element overhead and digital signatures that consist of a single group element. Notably, truncation collision resistance is also achieved by a non-programmable random oracle, even though this security notions is considered as a (non-standard, but seemingly reasonable) security notion for standard-model cryptographic hash functions. However, the main disadvantages of the constructions in [JK18] are that very strong computational hardness assumptions (so-called q-type assumptions with very large q) are required, and that the reductions are extremely non-tight.

1.1 Our contributions.

We introduce *blockwise partitioning* as a new approach to leverage the assumption that a cryptographic hash function is *weak near-collision resistant*. We will show that our technique yields more efficient and tighter constructions of identity-based encryption, based on lattices and on pairings, and a highly efficient new verifiable random function. We give a more detailed comparison between blockwise partitioning based on weak near-collision resistance and the results from [JK18] in Section 2.

Near-collision resistance. We informally say that a hash functions is weak near-collision resistant if the generic birthday attack is the fastest algorithm to find collisions on a fixed subset of the output bits. We for-

Schemes	$\lvert\mathsf{mpk}\rvert$ **# of** $\mathbb{Z}_q^{n\times m}$ **matr.**	$\lvert\mathsf{usk}\rvert, \lvert\mathsf{ct}\rvert$ **# of** $\mathbb{Z}_q^m$**vec.**	**LWE param** $1/\alpha$	**Reduction Cost**	**Remarks**
[CHKP10]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	$\tilde{\mathcal{O}}(n^{1.5})$	$\mathcal{O}(\varepsilon^{\nu+1}/Q^{\nu})^{\ddagger}$	
[ABB10b]+[Boy10]	$\mathcal{O}(\lambda)$	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(n^{5.5})$	$\mathcal{O}(\varepsilon^2/qQ)$	
[Yam16]	$\mathcal{O}(\lambda^{1/\mu})^{\dagger}$	$\mathcal{O}(1)$	$n^{\omega(1)}$	$\mathcal{O}(\varepsilon^{\mu+1}/kQ^{\mu})^{\dagger}$	
[ZCZ16]	$\mathcal{O}(\log Q)$	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(Q^2n^{6.5})$	$\mathcal{O}(\varepsilon/kQ^2)$	Q-bounded
[AFL16]*	$\mathcal{O}(\lambda/\log^2\lambda)$	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(n^6)$	$\mathcal{O}(\varepsilon^2/qQ)$	
[BL16]	$\mathcal{O}(\lambda)$	$\mathcal{O}(1)$	$\mathsf{superpoly}(n)$	$\mathcal{O}(\lambda)$	
[KY16]	$\mathcal{O}(\lambda^{1/\mu})^{\dagger}$	$\mathcal{O}(1)$	$\mathcal{O}(n^{2.5+2\mu})^{\dagger}$	$\mathcal{O}((\lambda^{\mu-1}\varepsilon^{\mu}/Q^{\mu})^{\mu+1})^{\dagger}$	Ring-based
[Yam17a] + $\mathsf{F}_{\mathsf{MAH}}$ §	$\mathcal{O}(\log^3\lambda)$	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(n^{11})$	$\mathcal{O}(\varepsilon^{\nu+1}/Q^{\nu})^{\ddagger}$	
[Yam17a] + $\mathsf{F}_{\mathsf{AFF}}$ §	$\mathcal{O}(\log^2\lambda)$	$\mathcal{O}(1)$	$\mathsf{poly}(\lambda)$	$\mathcal{O}(\varepsilon^2/k^2Q)$	Expensive offline phase
Sec. 3	$\mathcal{O}(\log\lambda)$	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(n^6)$	$\mathcal{O}(\varepsilon^2/t^2)$	

Table 1: Comparison of Adaptively Secure IBEs based on LWE in the standard model

We compare adaptively secure IBE schemes under the LWE assumption that do not use random oracles. We measure the size of ct and usk in the number of $\mathbb{Z}_q^m$ vectors and the size of mpk in the number of $\mathbb{Z}_q^{n\times m}$ matrices. Q, ε and t, respectively, denote the number of queries, the advantage against the security of the respective IBE, and the runtime of an adversary. We measure the reduction cost by the advantage of the algorithm solving the LWE problem that is constructed from the adversary against the IBE scheme. All reduction costs were computed using the technique of Bellare and Ristenpart [BR09].

† The constant $\mu \in \mathbb{N}$ can be chosen arbitrarily. However, the reduction cost degrades exponentially in μ and hence it should be chosen rather small.

‡ $\nu > 1$ is the constant satisfying $c = 1 - 2^{-1/\nu}$, where c is the relative distance of an underlying error correcting code. ν can be chosen arbitrarily close to one by choosing c closer to $1/2$ [Gol08]. However, this comes with larger public keys as shown in [JN19].

$*$ The authors also propose an additional scheme that we do not include, because it relies on much stronger complexity assumptions.

§ Yamada [Yam17a] provides two instantiations of his IBE, one based on a modified admissible hash function ($\mathsf{F}_{\mathsf{MAH}}$) and one based on affine functions ($\mathsf{F}_{\mathsf{AFF}}$). When Yamada's scheme is used with the second instantiation, the key generation and encryption need to compute the description of a branching program that computes the division. This makes the construction less efficient.

mally introduce weak-near collision resistance in Definition 1. It can be seen as a new variant of truncation collision resistance [JK18], which essentially captures the same intuition and therefore can be considered equally reasonable.

Near-collision resistance has been studied in several previous works, such as [BC04, BCJ$^+$05, PS14]. The Handbook of Applied Cryptography [MvOV96, Remark 9.22] lists near-collision resistance as a desired property of hash functions and a potential *certificational property*. Moreover, the sponge construction for hash functions, which SHA-3 is based on, has been shown to be indifferentiable from a random oracle [BDPV08] in a slightly idealized model, which immediately implies near-collision resistance of the sponge construction in this model. Since weak near-collision resistance is an even weaker property, we view it as a natural property of cryptographic hash functions.

Lattice-based IB-KEM. We apply our approach to construct a lattice-based IB-KEM with constant size ciphertexts and public keys of size $O(\log \lambda)$. This scheme has efficiency close to existing selectively-secure ones, which makes progress towards answering an open problem posed in Peikert's survey on lattice cryptography [Pei15] on the existence of adaptively-secure schemes whose efficiency is comparable to selectively-secure ones.

Our construction uses similar techniques to [DM14, Alp15], who were only able to obtain digital signature schemes, due to the fact that their approach is based on "confined guessing" [BHJ$^+$15], which is incompatible with standard security models for identity-based KEMs. We resolve this by implementing a similar technique with near-collision resistant hash functions. Our IB-KEM is based on the IBE scheme of Yamada [Yam17a] and is currently the most efficient one with only $\mathcal{O}(\log \lambda)$ many matrices in the public parameters and only very small constant factors in the Landau symbol. Compared to Yamada's scheme [Yam17a], our scheme enjoys smaller LWE parameters and a smaller master public key. Compared to the scheme of Zhang *et al.* [ZCZ16], which to the best of our knowledge is the only other scheme with a master public key of size $\mathcal{O}(\log \lambda)$, our scheme has a smaller concrete key size and, most importantly, much better LWE parameters. Furthermore, the scheme by Zhang *et al.* is based on cover-free sets and thus requires the number of queries to be known a priori. We compare our scheme with previous schemes in Table 1, which is based on the respective table by Yamada [Yam17a].

Pairing-based IB-KEM. We also construct two new variants of the pairing-based identity-based encryption schemes of Boneh and Boyen [BB04a] and Waters [Wat05]. In comparison to [BB04a] we achieve adaptive security instead of selective security. In comparison to [Wat05] we have public parameters of size $O(\log \lambda)$ instead of $O(\lambda)$. Security is based on the same algebraic complexity assumption as the original schemes plus weak near-collision resistance. The security analysis is also much simpler than in [Wat05] or the simplified proof by Bellare and Ristenpart [BR09] and does not require an "artificial abort" [Wat05]. To our best knowledge, this is the first adaptively-secure IBE scheme where ciphertexts consist only of *two* elements of a prime order algebraic group with logarithmic-size public parameters. The scheme also gives rise to an adaptively-secure (EUF-CMA) CDH-based digital signature scheme with logarithmic-size keys. See Table 2.

We also describe a new *adaptively-secure* variant of a scheme by Boneh and Boyen [BB04a] based on a q-type assumption where a ciphertext consists only of a *single* group element. In comparison to the corresponding construction from [JK18], the q of the required q-type assumption is reduced *quadratically*, while the tightness of the reduction is improved *quadratically*, too. This scheme also gives rise to a signature scheme with adaptive security in the standard model, where a signature is only a single element from a prime-order group, which achieves the same quadratic improvement over a construction from [JK18].

Scheme	$\|\mathsf{mpk}\|$	$\|\mathsf{usk}\|$	$\|\mathsf{ct}\|$	Security	Assumption	ROM	Security Loss
[BF01]	2	1	1	adap.	DBDH	Yes	$O(q_{key})$
[Wat05]	$O(\lambda)$	2	2	adap.	DBDH	No	$O(t^2 + (\lambda \cdot q_{key} \cdot \varepsilon^{-1})^2)$
[Wat09]	13	9	10	adap.	DLIN+DBDH	No	$O(q_{key})$
[Lew12]	25	6	6	adap.	DLIN	No	$O(q_{key})$
[CLL+13]	9	4	4	adap.	SXDH	No	$O(q_{key})$
[AHY15]	$O(\lambda)$	8	8	adap.	DLIN	No	$O(\log(\lambda))$
Sec. 4.1	$O(\log \lambda)$	2	2	adap.	DBDH	No	$O(t^2_{\mathcal{A}}/\varepsilon_{\mathcal{A}})$
[BB04a]	4	2	2	selec.	qDBDHI	No	$O(1)$
[Gen06]	3	2	3	adap.	qABDHE	No	$1 + \mathcal{O}(q^2_{key})/t_{\mathcal{A}}$
[JK18]	$O(\lambda)$	1	1	adap.	qDBDHI	No	$O(t^7_{\mathcal{A}}/\varepsilon^4_{\mathcal{A}})$
Sec. 4.2	$O(\lambda)$	1	1	adap.	qDBDHI	No	$O(t^3_{\mathcal{A}}/\varepsilon^2_{\mathcal{A}})$

Table 2: Comparison of IB-KEMs based on pairings with prime order groups and short ciphertexts. $|\mathsf{mpk}|$ is the number of group elements in public keys (descriptions of groups and hash functions not included), λ the security parameter. All public keys include at least one element from the target group of the pairing, except for [BF01]. $|\mathsf{usk}|$ and $|\mathsf{ct}|$ are the respective numbers of group elements in the user secret keys and ciphertexts when viewed as a KEM. "adap." means adaptive IND-ID-CPA security as defined below, "selec." is selective security in the sense of [BB04a]. The security loss is defined as the value L that satisfies $t_{\mathcal{B}}/\varepsilon_{\mathcal{B}} = L \cdot t_{\mathcal{A}}/\varepsilon_{\mathcal{A}}$, where $t_{\mathcal{A}}$,$\varepsilon_{\mathcal{A}}$ and $t_{\mathcal{B}}$,$\varepsilon_{\mathcal{B}}$ are the respective running time and advantage of the adversary and reduction and we ignored negligible terms in the security loss. q_{key} is the number of identity key queries.

Pairing-based VRF. As our last contribution, we construct a new VRF based on the q-DBDHI assumption by using blockwise partitioning and techniques of Yamada's VRF [Yam17a]. Our VRF is the first to achieve both small public keys and small proofs at the same time. Furthermore, the size of the keys and proofs is not only asymptotically small but also concretely: for $\lambda = 128$, public keys of our VRF consist of only 10 group elements and proofs of only 9 group elements. This is very close to the proof-size of Kohl's VRF [Koh19], which achieves proofs of size $\omega(1)$ but has larger public keys. Other constructions, like [Yam17a, Kat17] that also achieve (poly-)logarithmic size public keys and/or proofs suffer from large constant factors, even under very optimistic assumptions as shown by Jager and Niehues [JN19]. Furthermore, unlike other VRF constructions, we do not need the artificial abort technique [Wat05] or balancing [BR09, Jag15], leading to a simpler reduction. We compare our construction with previous constructions in Table 3, which is based on the respective table by Kohl [Koh19].

2 Blockwise Partitioning via Near-Collision Resistance

2.1 High-level approach.

Confined guessing [BHJ+13, BHJ+15] is a semi-generic technique to construct efficient and adaptively-secure digital signature schemes. It has been used for instance in [DM14, Alp15]. Unfortunately, it is only applicable to signatures, but neither to identity-based schemes such as identity-based key encapsulation mechanisms (IB-KEMs), nor to verifiable random functions (VRFs). The reason is essentially that this approach achieves only non-adaptive security, which is sufficient for signatures (as adaptive security can

Schemes	$\lvert\mathsf{vk}\rvert$	$\lvert\pi\rvert$	Assumption	Security loss
[HW10]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda \cdot Q)$-DDHE	$\mathcal{O}(\lambda Q/\varepsilon)$
[BMR10]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$-DDH	$\mathcal{O}(\lambda)$
[Jag15]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	$\mathcal{O}(\log(Q/\varepsilon))$-DDH	$\mathcal{O}(Q^\nu/\varepsilon^{\nu+1})$
[HJ16]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	DLIN	$\mathcal{O}(\lambda \log(\lambda) Q^{2/c}/\varepsilon^3)$
[Yam17a] Sec. 6.1	$\omega(\lambda \log^2 \lambda)^\dagger$	$\omega(\log^2 \lambda)^\dagger$	$\tilde{\mathcal{O}}(\lambda)$-DDH	$\mathcal{O}(Q^\nu/\varepsilon^{\nu+1})$
[Yam17a] Sec. 6.2	$\omega(\log^2 \lambda)^\dagger$	$\omega(\sqrt{\lambda} \log^2 \lambda)^\dagger$	$\tilde{\mathcal{O}}(\lambda)$-DDH	$\mathcal{O}(Q^\nu/\varepsilon^{\nu+1})$
[Yam17b] App. C.	$\omega(\log^2 \lambda)^\dagger$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$-DDH	$\mathcal{O}(\lambda^2 Q/\varepsilon^2)$
[Kat17] Sec. 5.1	$\omega(\log^2 \lambda)^\dagger$	$\omega(\lambda \log^2 \lambda)^\dagger$	$\omega(\log^2 \lambda)^\dagger$-DDH	$\mathcal{O}(Q^\nu/\varepsilon^{\nu+1})$
[Kat17] Sec. 5.3	$\omega(\sqrt{\lambda} \log \lambda)^\dagger$	$\omega(\log \lambda)^\dagger$	$\omega(\log^2 \lambda)^\dagger$-DDH	$\mathcal{O}(Q^\nu/\varepsilon^{\nu+1})$
[Ros18]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	DLIN	$\mathcal{O}(\lambda \log(\lambda) Q^{2/c}/\varepsilon^3)$
[Koh19]	$\omega(\lambda \log \lambda)^\dagger$	$\omega(\log \lambda)^\dagger$	DLIN	$\mathcal{O}(\lvert\pi\rvert \log(\lambda) Q^{2/\nu}/\varepsilon^3)$
[Koh19]	$\omega(\lambda^{2+2\eta})$	$\omega(1)^\dagger$	DLIN	$\mathcal{O}(\lvert\pi\rvert \log(\lambda) Q^{2+2/\nu}/\varepsilon^3)$
[JN19]	$\mathcal{O}(\lambda)$	$\mathcal{O}(\lambda)$	$\mathcal{O}(t^2/\varepsilon)$-DDH	$\mathcal{O}(t^3/\varepsilon^2)$
Sec. 5	$\mathcal{O}(\log \lambda)$	$\mathcal{O}(\log \lambda)$	$\mathcal{O}(t^2/\varepsilon)$-DDH	$\mathcal{O}(t^2/\varepsilon^2)$

Table 3: Comparison of Adaptively Secure VRFs in the standard model

We compare adaptively secure VRF schemes in the standard model. We measure the size of vk and π in the number of the respective group. Q, ε and t respectively denote the number of queries an adversary makes, the adversaries advantage against the security of the respective VRF and the adversaries runtime. Most of the constructions use an error correcting code $C : \{0,1\}^\lambda \to \{0,1\}^n$ with constant relative minimal distance $c \leq 1/2$, where n,$\nu > 1$ can be chosen arbitrarily close to 1 by choosing c arbitrarily close to $1/2$ [Gol08, Appendix E.1]. However, this leads to larger n and by that to larger public keys and/or proofs as shown in [JN19].

† Note that these terms only hold for "λ large enough" and therefore, key and proof sizes might have to be adapted with larger constants in order to guarantee adequate security.

then be easily achieved generically), but not for IB-KEMs or VRFs.

We propose *blockwise partitioning* as a new semi-generic technique, and show how it can be used to construct efficient IB-KEMs and VRFs with adaptive security. It is based on the *near-collision resistance* of a cryptographic hash function and similar in spirit to the closely related notion of *truncation collision resistance* [JK18].

High-level perspective. In order to sketch the main idea, consider IB-KEMs as an example. Our approach is to let the reduction guess $n' = \mathcal{O}(\log \lambda)$ many bits of $H(\mathsf{id}^*)$, where λ is the security parameter, H is a cryptographic hash function and id^* is the challenge identity chosen by the adversary. Using standard techniques from selectively-secure constructions, we prove security with a reduction which is successful if the n' bits of $H(\mathsf{id}^*)$ are guessed correctly, while the hash of every identity for which the adversary requests a secret key for differs in at least one of the n' bits.

For this approach to yield a reduction with non-negligible loss, we have to choose n' such that it fulfills the following two conflicting goals:

1. n' has to be small enough, such that the probability of guessing n' bits of $H(\mathsf{id}^*)$ correctly is non-negligible
2. n' has to be large enough to ensure that it is unlikely, relative to the adversary's advantage, to make a query id whose hash also matches on the n' guessed bits ("near-collision-resistance")

Following [JK18] (which in turn is inspired by [BHJ$^+$13, BHJ$^+$15]), we balance these two goals by choosing n' depending on the runtime and advantage of the adversary. This approach thus yields an ideal choice of n' for each adversary. However, [JK18] were not able to use this ideal value of n' directly, since they required to set n' to a power of two (or they would require larger public parameters of size $O(\lambda)$ instead of $O(\log \lambda)$). Increasing n' to a factor of almost two (in the worst case) incurs an additional quadratic security loss and also may requires a stronger q-type assumption with quadratically larger q.

We address this issue by viewing the output of the hash function as the concatenation of blocks of exponentially growing length, i.e. the first bit is the first block, bits two and three are the second block, bits four to seven are the third block, and so on. Our reduction then uses the ideal choice for n' and guesses the bits in the blocks whose lengths sum up to exactly n'. This more fine-grained guessing yields constructions with tighter security from weaker assumptions. It also reduces the required output length of the hash function from $4(\lambda+1)$ bits in [JK18] to only $2\lambda+3$ bits. Note that this is essentially optimal for a collision-resistant hash function. In particular, for many practical construction one would probably use a collision resistant hash function, anyway, to map long identities to short strings. We compare our techniques to the ones of [JK18] in more detail after formally introducing blockwise partitioning.

In the remainder of this section we will describe the framework and assumptions for blockwise partitioning, give some more technical intuition, and state and prove a technical lemma that will be useful to use blockwise partitioning as modular as possible in security proofs.

2.1.1 Blockwise partitioning.

Let $H : \{0,1\}^* \to \{0,1\}^n$ be a hash function. We will assume in the sequel that $n = \sum_{i=0}^{\ell} 2^i$ for simplicity and ease of exposition. One can generalize this to arbitrary n, but this would make the notation rather cumbersome without providing additional insight or clarity. Then we can view the output space $\{0,1\}^n$ of the hash function as a direct product of sets of exponentially-increasing size

$$\{0,1\}^n = \{0,1\}^{2^0} \times \cdots \times \{0,1\}^{2^\ell}.$$

For a hash function H we define functions $H_0, \ldots, H_\ell$ such that

$$H_i : \{0,1\}^* \to \{0,1\}^{2^i} \qquad \text{and} \qquad H(x) = H_0(x)||\cdots||H_\ell(x).$$

One can consider each $H_i(x)$ as one "block" of $H(x)$. Note that blocks have exponentially increasing size and there are $\lfloor \log n \rfloor + 1$ blocks in total.

Using blockwise partitioning. Let $t = t(\lambda)$ be a polynomial and let $\varepsilon = \varepsilon(\lambda)$ be a non-negligible function such that $\varepsilon > 0$ and $t/\varepsilon < 2^\lambda$ for all λ. Think of t and ε as (approximations of) the running time and advantage of an adversary in a security experiment. We define an integer n' depending on (t, ε) as

$$n' := \lceil \log(4t \cdot (2t-1)/\varepsilon) \rceil \tag{1}$$

Note that if $n \geq 2\lambda + 3$, then we have $0 \leq n' \leq n$ as we show in Lemma 1 below.

The value n' uniquely determines an index set $\mathcal{I} = \{i_1, \ldots, i_\omega\} \subseteq \{0, \ldots, \ell\}$ such that $n' = \sum_{i \in \mathcal{I}} 2^i$, where $\ell := \lfloor \log n \rfloor$. The key point in defining n' as in Equation (1) is that it provides the following two properties simultaneously:

Guessing a from polynomially-bounded range. In order to enable a reduction from adaptive to selective security, we will later have to "predict" a certain hash values $H(x^*)$. Think of x^* as the challenge identity in an IB-KEM security experiment, or the message from the forgery in a signature security experiment. Blockwise partitioning enables this as follows.

Consider the following probabilistic algorithm BPSmp, which takes as input λ, t, and ε, computes n' as in Equation (1), chooses $K_i \xleftarrow{\$} \{0,1\}^{2^i}$ uniformly random for $i \in \mathcal{I}$ and defines $K_i = \perp$ for all $i \notin \mathcal{I}$. Then it outputs

$$(K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t, \varepsilon).$$

The joint range of all hash functions H_i with $i \in \mathcal{I}$ is $\{0,1\}^{2^{i_1}} \times \cdots \times \{0,1\}^{2^{i_{|\mathcal{I}|}}}$, which has size

$$2^{n'} = 2^{\sum_{i \in \mathcal{I}} 2^i}.$$

Hence, we have that

$$\Pr\left[H_i(x^*) = K_i \text{ for all } i \in \mathcal{I}\right] = 2^{-n'}.$$

Note that $2^{n'}$ is *polynomially bounded*, due to the definition of n' in Equation (1).

Upper bound on the collision probability. In Lemma 1 below we will show that near-collision resistance of H guarantees that the probability that an adversary running in time t outputs any two values $x \neq x'$ such that

$$H_i(x) = H_i(x') \qquad \text{for all } i \in \mathcal{I} \tag{2}$$

is at most $\varepsilon/2$. Think of x and x' as values chosen adaptively by an adversary in a security experiment. In the context of IB-KEMs this would be chosen identities, in context of digital signatures chosen messages, for instance. Note that we do not argue that there is a *negligible* collision probability. This is not possible, because we consider a polynomially-bounded space, where an adversary will always be able to find collisions with non-negligible probability. However, we can guarantee that there will be no collision with probability at least $\varepsilon/2$. This means that an adversary that runs in some time t and has some advantage ε will sufficiently often be successful *without* finding a collision.

$\underline{n'\text{-}\mathsf{wNCR}^{\mathcal{H}}_{\mathcal{A}}}$

$(\mathcal{J}, st) \xleftarrow{\$} \mathcal{A}_1(n')$
$H \xleftarrow{\$} \mathcal{H}$
$(X^{(1)}, \ldots, X^{(Q+1)}) \xleftarrow{\$} \mathcal{A}_2(H, st)$
If $|\mathcal{J}| = n'$ and $\exists x \neq y \in \{X^{(1)}, \ldots, X^{(Q+1)}\}$ with $H(x)[i] = H(y)[i]$ for all $i \in \mathcal{J}$:
  return 1, else 0

Figure 1: The security experiment for weak near-collision resistance, executed with a family of hash functions $\mathcal{H}$ and adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1$ outputs an index set $\mathcal{J} \subseteq [n]$ and $\mathcal{H} \subseteq \{h : \{0,1\}^* \to \{0,1\}^n\}$. We restrict $\mathcal{A}_1$ to only output index sets $\mathcal{J}$ with $|\mathcal{J}| = n'$. Note that $H(x)[i]$ denotes the i-th bit of $H(x)$.

Hence, similar to confined guessing [BHJ+13, BHJ+15] and truncation collision resistance [JK18], blockwise partitioning enables us to guess challenge identities from a *polynomially bounded* space. At the same time, it ensures that the space is large enough such that collisions are sufficiently unlikely, such that any adversary breaking a considered cryptosystem with some advantage ε must "sufficiently often" be successful without finding a collision.

Blockwise partitioning via near-collision resistance. We will now give a formal definition of weak near-collision resistance and then provide a technical lemma, which will be useful for security proofs based on blockwise partitioning of hash function outputs. Note that weak near-collision resistance is only required for the security of our constructions and we hence only require this property in the respective theorems and not in the constructions themselves.

Definition 1 (Weak near-collision resistance)**.** Let $\mathcal{H} = \{H : \{0,1\}^* \to \{0,1\}^n\}$ be a family of hash functions. For $n' \in \{1, \ldots, n\}$, we say that an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ breaks the weak n'-near-collision resistance of $\mathcal{H}$, if it runs in time $t_{\mathcal{A}}$ and it holds that

$$\Pr\left[n'\text{-}\mathsf{wNCR}^{\mathcal{H}}_{\mathcal{A}} = 1\right] \geq t_{\mathcal{A}}(t_{\mathcal{A}} - 1)/2^{n'+1},$$

where n'-wNCR is the experiment defined in Figure 1 and the probability is over the randomness of $\mathcal{A}$ and choosing H. We say that $\mathcal{H}$ is *weak near-collision resistant*, if there exists no adversary $\mathcal{A}$ breaking the weak n'-near-collision resistance of $\mathcal{H}$ for any $n' \in \{1, \ldots, n\}$.

The following lemma will be useful to apply blockwise partitioning in security proofs.

Lemma 1. *Let $H : \{0,1\}^* \to \{0,1\}^n$ be a hash function, t be a polynomial, and let ε be a non-negligible function such that $\varepsilon > 0$ and $t/\varepsilon < 2^\lambda$ for all λ. Let $n' := \lceil \log(4t \cdot (2t-1)/\varepsilon) \rceil$ as in Equation* (1) *and define set $\mathcal{I}$ such that $n' = \sum_{i \in \mathcal{I}} 2^i$. Let $\mathcal{A}$ be an algorithm that outputs $(X^{(1)}, \ldots, X^{(Q)}, X^*)$ and runs in time t and let*

$$(K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t, \varepsilon),$$

where BPSmp *is the algorithm described above.*

1. *Let* coll *be the event that there exists $x, x' \in \{X^{(1)}, \ldots, X^{(Q)}, X^*\}$ such that*

$$H_i(x) = H_i(x') \text{ for all } i \in \mathcal{I}. \tag{3}$$

Let $\mathsf{badChal}$ *be the event that there exists* $i \in \mathcal{I}$ *such that* $\Pr[H_i(X^*) \neq K_i]$. *If* H *is drawn uniformly at random from a family of weak near-collision resistant hash functions in the sense of Definition 1, then we have*

$$(\varepsilon - \Pr[\mathsf{coll}]) \cdot \Pr[\neg\mathsf{badChal}] \geq \varepsilon^2/(32t^2 - 16t).$$

Moreover, coll *and* $\mathsf{badChal}$ *are independent of each other.*

2. *Let* $\mathsf{badEval}$ *be the event that there exists* $x \in \{X^{(1)}, \ldots, X^{(Q)}\}$ *with* $x \neq X^*$ *such that* $H_i(x) = K_i$ *for all* $i \in \mathcal{I}$. *Then we have*

$$\mathsf{badEval} \implies \mathsf{coll} \vee \mathsf{badChal}.$$

PROOF. The proof uses the following inequalities and identities from [JK18, JN19].

$$n' \in \{1, \ldots, 2k+3\}, \qquad \frac{2t(2t-1)}{2^{n'}} \leq \frac{\varepsilon}{2} \qquad \text{and} \qquad \frac{1}{2^{n'}} \geq \frac{\varepsilon}{16t^2 - 8t} \tag{4}$$

The proof for these inequalities is given in Section A for completeness. We start to prove Property 1 by showing $\Pr[\mathsf{coll}] < \varepsilon/2$. Assume an algorithm $\mathcal{A}$ running in time $t_{\mathcal{A}}$ that outputs $(X^{(1)}, \ldots, X^{(Q)}, X^*)$ such that there exist $x, x' \in \{X^{(1)}, \ldots, X^{(Q)}, X^*\}$ such that Equation (3) holds with probability at least $\varepsilon/2$. By the definition of $\mathcal{I}$ and the functions H_i, this yields that $H(x)$ and $H(x')$ agree on at least n' positions. We construct an algorithm $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ that uses $\mathcal{A}$ to break the weak n'-near-collision resistance of $\mathcal{H}$. Note that the choice of $\mathcal{I}$ is independent of $H \in \mathcal{H}$. $\mathcal{B}_1$ therefore just encodes $K = (K_0, \ldots, K_\ell)$ to $\mathcal{J} \subseteq \{1, \ldots, n\}$ with $|\mathcal{J}| = n'$. $\mathcal{B}_2$ simply relays $\mathcal{A}$'s output $(X^{(1)}, \ldots, X^{(Q)}, X^*)$. The runtime $t_{\mathcal{B}}$ of $\mathcal{B}$ is at most $2t_{\mathcal{A}}$, since $\mathcal{B}$ does nothing more than executing $\mathcal{A}$ and relaying its outputs. Therefore, we get

$$\Pr[\mathsf{coll}] > \varepsilon_{\mathcal{A}}/2 \geq \frac{2t_{\mathcal{A}}(2t_{\mathcal{A}} - 1)}{2^{n'}} \geq \frac{t_{\mathcal{B}}(t_{\mathcal{B}} - 1)}{2^{n'+1}},$$

where the second inequality follows from Equation (4). This contradicts the weak near-collision resistance of $\mathcal{H}$. Next, we determine $\Pr[\neg\mathsf{badChal}]$. We have that the events coll and $\mathsf{badChal}$ are independent of each other because $(K_0, \ldots, K_\ell)$ is chosen independently from $(X^{(1)}, \ldots, X^{(Q)}, X^*)$. Moreover, each K_i with $i \in \mathcal{I}$ is chosen uniformly at random from $\{0,1\}^{2^{2^i}}$ and thus we have

$$\Pr[\neg\mathsf{badChal}] = \Pr[H_i(X^*) = K_i \text{ for all } i \in \mathcal{I}] = \frac{1}{2^{\sum_{i \in \mathcal{I}} 2^i}} = 2^{-n'},$$

where the last equation follows by definition of n'. To prove Property 1, we then calculate

$$(\varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}])2^{-n'} \geq \left(\varepsilon_{\mathcal{A}} - \frac{\varepsilon_{\mathcal{A}}}{2}\right) \frac{\varepsilon_{\mathcal{A}}}{16t_{\mathcal{A}}^2 - 8t_{\mathcal{A}}} = \frac{\varepsilon_{\mathcal{A}}^2}{32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}},$$

where the first inequality follows from Equation (4). Finally, to show Property 2, we explain that if $\mathsf{badEval}$ occurs, then either $\mathsf{badChal}$ or coll *must occur*. This is because if there exists $x \in \{X^{(1)}, \ldots, X^{(Q)}\}$ with $x \neq X^*$ and $H_i(x) = K_i$ for all $i \in \mathcal{I}$, then we have either that also $H_i(X^*) = K_i$ for all $i \in \mathcal{I}$ and then coll occurs or we have that there exists an index $i \in \mathcal{I}$ such that $H_i(X^*) \neq K_i$ and then $\mathsf{badChal}$ occurs. This concludes the proof. □

Near-collision resistance and the non-programmable random oracle model. Near-collision resistance holds *unconditionally* in the non-programmable random oracle model [FLR+10]. Hence, all our results can also be viewed as a generic technique to obtain adaptively-secure cryptosystems in the non-programmable random oracle model without any additional assumptions. In this sense, our paper is in line with recent works that aim to avoid programmability, such as [FHJ20].

Comparison to ELFs. Extremely lossy functions (ELFs), which were introduced by Zhandry in [Zha16], are hash functions that allow the reductions to choose the hash function's image size depending on the adversary, such that a function with a small image size is indistinguishable from an injective hash function. Blockwise partitioning uses the weak near-collision resistance of standard hash functions in a similar manner, by selecting the blocks depending on the adversary's runtime and advantage. Hence, ELFs have the potential to enable constructions similar to the ones we present. However, the known construction based on exponential hardness of the DDH problem relies on public key techniques and thus is less efficient then a standard hash functions. Blockwise partitioning can also be seen as an approach towards resolving the open problem of constructing ELFs from symmetric-key primitives. While we syntactically do not construct an ELF, the way blockwise partitioning is used in a proof is very similar.

Comparison to confined guessing and truncation collision resistance. Note that the index set $\mathcal{I}$ defined above may contain *multiple* indices. This is a major difference of our approach to confined guessing and truncation collision resistance, where always only *single* blocks are guessed.

The advantage of being able to guess multiple blocks is that we are now able to define n' in a much more fine-grained way, as *any* integer between 0 and n. In contrast, [BHJ$^+$13, BHJ$^+$15] and [JK18] were only able to pick values n' of exponentially increasing size, such that $n' = 2^{2^j}$ for some j, which is the reason why our reductions can improve tightness and the strength of the required assumptions quadratically.

However, we cannot replace the approach of [BHJ$^+$13, BHJ$^+$15] and [JK18] with blockwise partitioning in a black-box manner. Instead, we have to provide a new security analysis for cryptosystems, and show that there are reductions which are compatible with guessing multiple blocks.

3 Lattice-based IB-KEM

We describe how blockwise partitioning can be applied in the context of lattice based cryptography, using an *Identity-Based Key-Encapsulation-Mechanism* (IB-KEM) based on LWE as example. We build our IB-KEM from Yamada's IBE [Yam17a], for which we describe how blockwise partitioning can be embedded into lattice trapdoors by describing "compatible algorithms" for blockwise partitioning in the lattice context. The notion is inspired by [Yam17a] and we use it as a generic building block to instantiate the IB-KEM. The instantiation then has ciphertexts and secret keys consisting of a constant number of matrices and vectors and public keys consisting of only $\mathcal{O}(\log(\lambda))$ many matrices and vectors. Furthermore, we are able to achieve better LWE-parameters. We start by providing preliminaries on lattices, define compatible algorithms and our instantiation of them based on blockwise partitioning before finally presenting the construction.

3.1 Preliminaries on Lattices

For an integer $m > 0$, let $D_{\mathbb{Z}^m,\sigma}$ be the discrete Gaussian distribution over $\mathbb{Z}^m$ with parameter $\sigma > 0$. Further, for $q, n \in \mathbb{Z}$, $\mathbf{u} \in \mathbb{Z}_q^m$ and $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, the m-dimensional integer lattices $\Lambda_p^{\perp}(\mathbf{A})$ and $\Lambda_p^{\mathbf{u}}(\mathbf{A})$ are defined as follows

$$\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m : \mathbf{A}\mathbf{e} = 0 \bmod q\}, \qquad \Lambda_q^{\mathbf{u}}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m : \mathbf{A}\mathbf{e} = \mathbf{u} \bmod q\}.$$

Throughout the section we will always assume $n = \Theta(\lambda)$. In particular, any function negligible in n is therefore also negligible in λ. Moreover, we denote the ℓ_2-norm by $\|\cdot\|_2$ and the infinity norm by $\|\cdot\|_\infty$. Namely, that is $\|\mathbf{A}\|_\infty = \max_{\substack{1\le i\le n\\ 1\le j\le m}} |a_{i,j}|$. Then the following lemma holds.

Lemma 2 ([Reg05]). *We have* $\Pr\left[\|x\|_2 > \sigma\sqrt{m} : \mathbf{x} \xleftarrow{\$} D_{\mathbb{Z}^m,\sigma}\right] \leq 2^{-2m}$.

The *learning with errors* (LWE) problem was defined by Regev [Reg05].

Definition 2 ([Reg05]). For integers $n = n(\lambda), m = m(\lambda)$, 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

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

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 the $\mathsf{dLWE}_{n,m,q,\alpha}$ assumption holds if $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\alpha}}(\lambda)$ is negligible in λ for all PPTs $\mathcal{A}$.

Regev also showed that solving $\mathsf{dLWE}_{n,m,q,\alpha}$ for $\alpha q \geq 2\sqrt{n}$ is (quantumly) at least as hard as solving GapSVP_γ and SIVP_γ on arbitrary n-dimensional lattices for some $\gamma = \tilde{\mathcal{O}}(n/\alpha)$ [Reg05]. Later on, partial dequantization of Regev's reduction was achieved [Pei09, BLP+13]. Furthermore, Katsumata and Yamada introduced the following lemma that allows to rerandomize LWE instances.

Lemma 3 (Lemma 1 in [KY16]). *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 $\mathbf{x} \xleftarrow{\$} D_{\mathbb{Z}^m,r}$. 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 $\mathbf{b}^T\mathbf{V} + \mathbf{x'}^T \in \mathbb{Z}_q^{m'\times 1}$, where $\mathbf{x}'$ is distributed statistically close to $D_{\mathbb{Z}^{m'},2rs}$.*

Trapdoors. We use two types of trapdoors, trapdoors using a short basis for the lattice and "gadget"-based trapdoors. We start by introducing the former.

Following [BV16], we let $n, m, m', q \in \mathbb{N}$ 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 $\mathbf{A}_\sigma^{-1}(\mathbf{V})$ denote the random variable whose distribution 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 $\mathbf{V} \in \mathbb{Z}_q^{n,\times m'}$. Note that we slightly overload notation and denote a σ-trapdoor for $\mathbf{A}$ by $\mathbf{A}_\sigma^{-1}$.

Lemma 4. *The following properties have been established [GPV08, ABB10a, ABB10b, CHKP10, BLP+13].*

1. *There exists an efficient algorithm $\mathsf{GenTrap}(1^n, 1^m, q)$ that, for some $m = \mathcal{O}(n\log(q))$, outputs $\left(\mathbf{A}, \mathbf{A}_{\sigma_0}^{-1}\right)$ such that $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is 2^{-n}-close to a uniformly random matrix and we have $\sigma_0 = \omega(\sqrt{n\log q\log m})$.*
2. *Given $\mathbf{A}_\sigma^{-1}$, one can obtain $\mathbf{A}_{\sigma'}^{-1}$ for any $\sigma' \geq \sigma$.*
3. *Given $\mathbf{A}_\sigma^{-1}$, one can obtain $[\mathbf{A} \mid \mathbf{B}]_\sigma^{-1}$ for any $\mathbf{B}$.*
4. *For $\mathbf{A}_\sigma^{-1}$ and $\mathbf{u} \in \mathbb{Z}_q^n$, it holds $\Pr\left[\left\|\mathbf{A}_\sigma^{-1}\right\|_2 > \sqrt{m}\cdot\sigma\right] = \mathsf{negl}(\lambda)$.*

We will also use a second type of trapdoors known as "gadget"-based trapdoors introduced by Micciancio and Peikert [MP12].

Lemma 5 (Theorem 1 from [MP12]). *Let $m \geq n\lceil\log(q)\rceil$, then there is a fixed "gadget" matrix $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$, such that $\mathbf{G}$ has full rank and there is an efficient algorithm $\mathbf{G}^{-1}$ that on input a matrix $\mathbf{U} \in \mathbb{Z}_q^{n\times m}$ outputs $\mathbf{V} \in \{0,1\}^{m\times m}$ such that $\mathbf{GV} = \mathbf{U}$. Note that we slightly abuse notation here by writing $\mathbf{G}^{-1}$ even though $\mathbf{G}^{-1}$ is not the inverse of $\mathbf{G}$ but an efficient algorithm.*

These trapdoors have some further useful properties described below.

Lemma 6 (Section 3.4 in [MP12]). *Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{R} \in \mathbb{Z}_q^{m\times m}$ and $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$ an invertible matrix, then we can efficiently sample from $[\mathbf{A}|\mathbf{AR}+\mathbf{HG}]_\sigma^{-1}$ for $\sigma = m \cdot \|\mathbf{R}\|_\infty \cdot \omega(\sqrt{\log(m)})$.*

When using the trapdoors, we will need that $(\mathbf{A}, \mathbf{AR}+\mathbf{HG})$ is negligibly close to $(\mathbf{A}, \mathbf{A}')$ for random matrices $\mathbf{A}$ and $\mathbf{A}'$. This can be guaranteed by the *leftover hash lemma*.

Lemma 7 (Leftover hash lemma [Reg05, ABB10a]). *Let $m \geq (n+1)(\log(q)+\omega(\log(n))$, then for $\mathbf{R} \xleftarrow{\$} \{-1,1\}^{m\times m}$, $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ it holds that the statistical difference between the distributions $(\mathbf{A}, \mathbf{AR})$ and $(\mathbf{A}, \mathbf{A}')$ is negligible in n.*

3.2 Compatible Algorithms

Our construction is based on $\mathsf{dLWE}_{n,m+1,q,\alpha}$. The construction follows Yamada's construction of a lattice IBE [Yam17a] and requires "compatible algorithms" to be instantiated. We first define properties required from these compatible algorithms and provide our instantiation of them based on blockwise partitioning in the next section.

Defining compatible algorithms Let $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ be the gadget matrix as introduced in [MP12, Theorem 1]. That is, $\mathbf{G}$ is a full rank matrix for which there is an efficient algorithm $\mathbf{G}^{-1}$ that on input $\mathbf{U} \in \mathbb{Z}_q^{n\times m}$ outputs a matrix $\mathbf{V} \in \{-1,1\}^{m\times m}$ such that $\mathbf{GV} = \mathbf{U}$. We provide a formal definition of $\mathbf{G}$ in Section 3.1. We then say that the algorithms Encode, $\mathsf{PubEval}$ and $\mathsf{TrapEval}$ are *compatible with blockwise partitioning* if they combine the *vanishing trapdoors technique* from [ABB10a, Boy10] with blockwise partitioning. That is, that Encode encodes $(K_0,\dots,K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t(\lambda), \varepsilon(\lambda))$ into matrices $\mathbf{B}, (\mathbf{B}_i)_{0\leq i\leq \ell}$ and trapdoors $\mathbf{R}, (\mathbf{R})_{0\leq i\leq \ell}$
such that $\mathsf{PubEval}(H, \mathsf{id}, \mathbf{B}, (\mathbf{B}_i)_{0\leq i\leq \ell})$ computes $\mathbf{B}_{\mathsf{id}}$ with

$$\mathbf{B}_{\mathsf{id}} = \begin{cases} \mathbf{AR}_{\mathsf{id}} + \mathbf{H}_{\mathsf{id}}\mathbf{G} & \text{if } H_i(\mathsf{id}) = K_i \text{ for all } i \in \mathcal{I} \\ \mathbf{AR}_{\mathsf{id}} & \text{otherwise,} \end{cases}$$

where $\mathbf{R}_{\mathsf{id}}$ is a matrix of small maximum norm that can be computed from the trapdoors using $\mathsf{TrapEval}$ and $\mathbf{H}_{\mathsf{id}}$ is a invertible matrix that depends on id. Note that we denote the infinity norm of a matrix $\mathbf{R}$ by $\|\mathbf{R}\|_\infty$.

Given these properties, the reduction can generate user secret keys for all identities id with $H_i(\mathsf{id}) \neq K_i$ for some $i \in \mathcal{I}$ by using a gadget trapdoor described in Section 3.1. At the same time, if id^* is such that $H_i(\mathsf{id}^*) = K_i$ for all $i \in \mathcal{I}$, then the reduction can extract a solution to its LWE instance using the adversary. By this, compatible algorithms allow us to apply blockwise partitioning in the context of lattices. We formally define these conditions as follows.

Definition 3. We say that the algorithms $(\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval})$ are *δ-compatible with blockwise partitioning using a family of hash functions $\mathcal{H}$*, if they are efficient and for all $\lambda \in \mathbb{N}, t = t(\lambda) = \mathsf{poly}(\lambda)$ and $\varepsilon = \varepsilon(\lambda)$ non-negligible in λ with $t(\lambda)/\varepsilon(\lambda) \leq 2^\lambda$, they satisfy the following properties:

- For some matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $(K_i)_{0\leq i\leq \ell} \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t(\lambda), \varepsilon(\lambda))$ it holds that $((\mathbf{B}, \mathbf{R}), (\mathbf{B}_i, \mathbf{R}_i)_{0\leq i\leq \ell}) = \mathsf{Encode}(\mathbf{A}, (K_i)_{0\leq i\leq \ell})$ with $\mathbf{B}, \mathbf{B}_i \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{R}, \mathbf{R}_i \in \{-1,1\}^{m\times m}$ for all $1 \leq i \leq \ell$.

- For $H \in \mathcal{H}$, $\mathsf{id} \in \{0,1\}^*$ and $(\mathbf{B}, (\mathbf{B}_i)_{0\le i\le \ell})$ with $\mathbf{B}_i \in \mathbb{Z}_q^{n\times m}$ for all $0 \le i \le \ell$ it holds that $\mathsf{PubEval}(H, \mathsf{id}, \mathbf{B}, (\mathbf{B}_i)_{0\le i\le \ell}) = \mathbf{B}_{\mathsf{id}} \in \mathbb{Z}_q^{n\times m}$.
- For $H \in \mathcal{H}, \mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{R}_i \in \mathbb{Z}_q^{m\times m}$ for all $0 \le i \le \ell$, and all $\mathsf{id} \in \{0,1\}^*$ it holds that $\mathsf{TrapEval}(H, \mathsf{id}, \mathbf{R}, (\mathbf{R}_i)_{0\le i\le \ell}) = \mathbf{R}_{\mathsf{id}} \in \mathbb{Z}^{m\times m}$.

We require that for all $\mathsf{id} \in \{0,1\}^*, \mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $H \in \mathcal{H}$ it holds that

$$\mathsf{PubEval}(H, \mathsf{id}, (\mathbf{B}_i)_{0\le i\le \ell}) \begin{cases} \mathbf{A}\mathbf{R}_{\mathsf{id}} & \text{if } H_i(\mathsf{id}) = K_i \text{ for all } i \in \mathcal{I} \\ \mathbf{A}\mathbf{R}_{\mathsf{id}} + \mathbf{H}_{\mathsf{id}}\mathbf{G} & \text{otherwise} \end{cases}$$

for some invertible matrix $\mathbf{H}_{\mathsf{id}} \in \mathbb{Z}_q^{n\times n}$ and that

$$\|\mathbf{R}_{\mathsf{id}}\|_\infty \le \delta,$$

where $(K_i)_{0\le i\le \ell}$ is sampled as $(K_i)_{0\le i\le \ell} \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t, \varepsilon)$ and we have that $\mathsf{Encode}(\mathbf{A}, (K_i)_{0\le i\le \ell}) = ((\mathbf{B}, \mathbf{R}), (\mathbf{B}_i, \mathbf{R}_i)_{0\le i\le \ell})$. Further $\mathbf{R}_{\mathsf{id}}$ is computed as $\mathbf{R}_{\mathsf{id}} = \mathsf{TrapEval}(H, \mathsf{id}, \mathbf{R}, (\mathbf{R}_i)_{0\le i\le \ell})$. Finally, we require, that for $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and all $0 \le i \le \ell$ the distributions $(\mathbf{A}, \mathbf{A}')$ and $(\mathbf{A}, \mathbf{B}_i)$ and the distributions $(\mathbf{A}, \mathbf{A}')$ and $(\mathbf{A}, \mathbf{B})$ have only negligible statistical difference in λ.

3.3 Instantiating Compatible Algorithms from Blockwise Partitioning

In this section we describe the main technical novelty of our lattice based construction: how blockwise partitioning can be applied in the context of lattices. We first discuss how a hash function output $H_i(X)$ is encoded as a matrix using the full-rank-difference encoding from Agrawal *et al.* [ABB10a] and adapt it to our needs. We then proceed to describe compatible algorithms using this encoding that fulfill all requirements of Definition 3 and can thus be used to instantiate our IB-KEM.

Encoding identities as full rank difference matrices. In our construction, we will first hash each $\mathsf{id} \in \{0,1\}^*$ with a weak near-collision resistant hash function $H \xleftarrow{\$} \mathcal{H}$ and then encode each $H_i(\mathsf{id})$ as an invertible matrix as described by Agrawal *et al.* [ABB10a]. In the following, we define the *full rank difference encoding function* of [ABB10a] and show how it can be adopted to fit blockwise partitioning. Informally, for a binary string $a \in \{0,1\}^{2^i}$, meaning a is a potential output of H_i, we pad a with zeros to be of length n by first padding it $\sum_{j=0}^{i-1} 2^j$ zeros in the front and with $\sum_{j=i+1}^{\ell} 2^j$ zeros in the end. We then canonically interpret it as a vector in $\mathbb{Z}_q^n$ and encode it with the full-rank difference encoding of [ABB10a]. We formalize this process in the following definition.

Definition 4. Let $f(\mathsf{Z})$ be an irreducible polynomial of degree n in $\mathbb{Z}_q^n[\mathsf{Z}]$ and for $\mathbf{a} \in \mathbb{Z}_q^n$, let $g_a(\mathsf{Z}) := \sum_{k=0}^{n-1} a_{k+1}\mathsf{Z}^k \in \mathbb{Z}_q^n[\mathsf{Z}]$. Then the function $\mathsf{FRD}(\mathbf{a}) : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n\times n}$ from [ABB10a] is defined as

$$\mathsf{FRD}(\mathbf{a}) := \begin{bmatrix} \mathsf{coeffs}(g_{\mathbf{a}} \bmod f) \\ \mathsf{coeffs}(\mathsf{Z} \cdot g_{\mathbf{a}} \bmod f) \\ \mathsf{coeffs}(\mathsf{Z}^2 \cdot g_{\mathbf{a}} \bmod f) \\ \vdots \\ \mathsf{coeffs}(\mathsf{Z}^{n-1} \cdot g_{\mathbf{a}} \bmod f) \end{bmatrix} \in \mathbb{Z}_q^{n\times n},$$

where coeffs denotes the coefficients of a polynomial in $\mathbb{Z}_q^n[\mathsf{Z}]$. For all $0 \le i \le \ell$ we define $\mathsf{FRD}_i : \{0,1\}^{2^i} \to \mathbb{Z}_q^{n\times n}$ to be the function that behaves as follows.

1. For an input $(a_1, \ldots, a_{2^i}) \in \{0,1\}^{2^i}$, FRD_i lets $\mathsf{offset}_i := \sum_{j=0}^{i-1} 2^j$ and sets $\mathbf{b}^T := [b_1, \ldots, b_n] \in \mathbb{Z}_q^n$, where
$$b_k := \begin{cases} a_{k-\mathsf{offset}_i} & \text{if } \mathsf{offset}_i < k \leq \mathsf{offset}_i + 2^i \\ 0 & \text{otherwise} \end{cases}$$
for all $1 \leq k \leq n$.

2. It then outputs $\mathsf{FRD}_i(a) := \mathsf{FRD}(\mathbf{b})$.

Agrawal *et al.* [ABB10a] prove some properties of FRD that immediately imply the following properties of FRD_i.

Lemma 8 (Section 5 in [ABB10a])**.** *Let $\mathsf{FRD}_i : \{0,1\}^{2^i} \to \mathbb{Z}_q^{n \times n}$ be as defined in Definition 4, then the following holds:*

1. *FRD_i is injective.*
2. *There is an additive group $\mathbb{G} \subset \mathbb{Z}_q^{n \times n}$ such that each $\mathbf{H} \in \mathbb{G} \setminus \{0\}$ is invertible and the range of FRD_i is a subset of $\mathbb{G}$ for all $1 \leq i \leq \ell$.*

We refer to [ABB10a, Section 5] for the proofs of the underlying facts used in Lemma 8. Our definition of FRD_i serves some further purposes that allows us to use it in conjunction with blockwise partitioning. We detail these properties in the following lemma.

Lemma 9. *Let BPSmp be as defined in Section 2 and let $t \in \mathbb{N}, \varepsilon \in (0,1]$ with $t/\varepsilon < 2^\lambda$. Then for $(K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t, \varepsilon)$, $\mathcal{I} = \{i : K_i \neq \perp\} \subseteq \{0, \ldots, \ell\}$ and $X \in \{0,1\}^*$ it holds that*
$$-\left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\right) + \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(H_i(X))\right) = 0 \Leftrightarrow K_i = H_i(X) \text{ for all } i \in \mathcal{I}.$$

PROOF. First, we observe that if $H_i(X) = K_i$ for all $i \in \mathcal{I}$, then it holds that
$$-\left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\right) + \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(H_i(X))\right) = -\left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\right) + \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\right) = 0,$$
which proves the first direction of the equivalence. We prove the second direction by contradiction. Informally, we do so by observing that the first row of $\mathbf{H}_X$ consists of the differences between K_i and $H_i(X)$ over $\mathbb{Z}_q$ for all $i \in \mathcal{I}$. Hence, $\mathbf{H} = 0$ implies $H_i(X) = K_i$ for all $i \in \mathcal{I}$, which contradiction the original assumption. We proceed by formalizing this proof by contradiction as follows.

Assume that there exists an $i^* \in \mathcal{I}$ such that $H_{i^*}(X) \neq K_i$ and $\mathbf{H}_X := -\left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\right) + \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(H_i(X))\right) = 0$ at the same time. Now for $i \in \mathcal{I}, 1 \leq j \leq n$ we denote the j-th element of the first row of $\mathsf{FRD}_i(H_i(X))$ by $b_{i,j} \in \mathbb{Z}_q$. Analogously, we denote the j-th element of the first row of $\mathbf{H}_X$ by $h_j \in \mathbb{Z}_q$. We now make the following observations that follow immediately from the definition of FRD_i in Definition 4.

1. For all $i \in \mathcal{I}$ and for all $j \in \{1, \ldots, n\} \setminus \{\mathsf{offset}_i + 1, \ldots, \mathsf{offset}_i + 2^i\}$ we have $b_{i,j} = 0$.
2. It holds $(b_{i,\mathsf{offset}_i+1}, \ldots, b_{i,\mathsf{offset}_i+2^i}) = H_i(X)$ and $(h_{\mathsf{offset}_i+1}, \ldots, h_{\mathsf{offset}_i+2^i}) = K_i$ for all $i \in \mathcal{I}$, where offset_i is as defined in Definition 4.

Combining the two observations yields that for all $i \in \mathcal{I}$ and $\mathsf{offset}_i + 1 \leq j \leq \mathsf{offset}_i + 2^i$ it holds that $[h_{\mathsf{offset}_i+1}, \ldots, h_{\mathsf{offset}_i+2^i}]^T = H_i(X) - K_i$, where we interpret $H_i(X)$ and K_i as vectors in $\mathbb{Z}_q^{2^i}$. Recall that by our assumption above, it holds that $\mathbf{H}_X = 0$ and hence $H_{i^*}(X) - K_{i^*} = 0$, if interpreted as vectors in $\mathbb{Z}_q^{2^i}$. This however contradicts $H_{i^*}(X) \neq K_i$, which proves the lemma. □

Next, we describe the algorithms $(\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval})$ and how they use FRD_i. Afterwards, we prove that the algorithms are compatible and can thus be used in our IB-KEM. The algorithms behave as follows:

$\mathsf{Encode}(\mathbf{A}, K_0, \ldots, K_\ell)$**:** The algorithm samples $\mathbf{R}, \mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$ for all $0 \leq i \leq \ell$ and sets

$$\mathbf{B}_i := \begin{cases} \mathbf{A}\mathbf{R}_i + \mathbf{G} & \text{if } K_i \neq \perp \\ \mathbf{A}\mathbf{R}_i & \text{if } K_i = \perp \end{cases}$$

and $\mathbf{B} := \mathbf{A}\mathbf{R} - \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\mathbf{G}\right)$. It then outputs the matrices $((\mathbf{B}, \mathbf{R}), (\mathbf{B}_i, \mathbf{R}_i)_{0 \leq i \leq \ell})$.

$\mathsf{PubEval}(H, \mathsf{id}, \mathbf{B}, (\mathbf{B}_i)_{0, \leq i \leq \ell})$**:** The algorithm computes $\mathbf{H}_i := \mathsf{FRD}_i(H_i(\mathsf{id}))$ for all $0 \leq i \leq \ell$ and sets $\mathbf{B}'_i := \mathbf{B}_i\mathbf{G}^{-1}(\mathbf{H}_i\mathbf{G})$. It then outputs $\mathbf{B}_{\mathsf{id}} := \mathbf{B} + \sum_{i=0}^{\ell} \mathbf{B}'_i$.

$\mathsf{TrapEval}(H, \mathsf{id}, \mathbf{R}, (\mathbf{R}_i)_{0 \leq i \leq \ell})$**:** The algorithm computes $\mathbf{H}_i := \mathsf{FRD}_i(H_i(\mathsf{id}))$ for all $0 \leq i \leq \ell$ and sets $\mathbf{R}'_i := \mathbf{R}_i\mathbf{G}^{-1}(\mathbf{H}_i\mathbf{G})$. It then outputs $\mathbf{R}_{\mathsf{id}} := \mathbf{R} + \sum_{i=0}^{\ell} \mathbf{R}'_i$.

Lemma 10. *The algorithms* $(\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval})$ *above are* $\delta = 1 + (\ell + 1)m$*-compatible with blockwise partitioning using the family of weak near-collision resistant hash functions* $\mathcal{H}$ *described in Section 2.*

PROOF. We first observe that the algorithms described above fulfill the syntactical requirements. We next show that

$$\mathsf{PubEval}(H, \mathsf{id}, (\mathbf{B}_i)_{0 \leq i \leq \ell}) \begin{cases} \mathbf{A}\mathbf{R}_{\mathsf{id}} & \text{if } H_i(\mathsf{id}) = K_i \text{ for all } i \in \mathcal{I} \\ \mathbf{A}\mathbf{R}_{\mathsf{id}} + \mathbf{H}_{\mathsf{id}}\mathbf{G} & \text{otherwise} \end{cases}$$

for some invertible matrix $\mathbf{H}_{\mathsf{id}} \in \mathbb{Z}_q^{n \times n}$. For $\mathbf{H}_i := \mathsf{FRD}_i(H_i(\mathsf{id}))$ and $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + x_i\mathbf{H}\mathbf{G}$, where $x_i = 1$ if $i \in \mathcal{I}$ and 0 otherwise, we observe that $\mathbf{B}'_i = \mathbf{B}_i\mathbf{G}^{-1}(\mathbf{H}_i\mathbf{G}) = \mathbf{A}\mathbf{R}_i\mathbf{G}^{-1}(\mathbf{H}_i\mathbf{G}) + x_i \cdot \mathbf{H}_i\mathbf{G} = \mathbf{A}\mathbf{R}'_i + x_i\mathbf{H}_i\mathbf{G}$, where $\mathbf{R}'_i$ is as defined by $\mathsf{TrapEval}$. We then have that

$$\begin{aligned} \mathbf{B}_{\mathsf{id}} = \mathbf{B} + \sum_{i=0}^{\ell} \mathbf{B}'_i &= \mathbf{A}\mathbf{R} - \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\mathbf{G}\right) + \left(\sum_{i=0}^{\ell} \mathbf{A}\mathbf{R}'_i + x_i\mathbf{H}_i\mathbf{G}\right) \\ &= \mathbf{A}\left(\mathbf{R} + \sum_{i=0}^{\ell} \mathbf{R}'_i\right) - \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\mathbf{G}\right) + \left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(H_i(\mathsf{id}))\mathbf{G}\right) \\ &= \mathbf{A}\mathbf{R}_{\mathsf{id}} - \mathbf{H}_{\mathsf{id}}\mathbf{G}, \end{aligned}$$

where $\mathbf{R}_{\mathsf{id}}$ is as in the description of $\mathsf{TrapEval}$ and $\mathbf{H}_{\mathsf{id}} = -\left(\sum_{i \in \mathcal{I}} \mathsf{FRD}_i(K_i)\right) + \left(\sum_{i \in \mathcal{I}} \mathbf{H}_i\right)$. Observe that $\mathbf{H}_{\mathsf{id}} = 0$ is equivalent to $K_i = H_i(\mathsf{id})$ for all $i \in \mathcal{I}$ by Lemma 9. Furthermore, we have by Lemma 8 that if $\mathbf{H}_{\mathsf{id}} \neq 0$, then $\mathbf{H}_{\mathsf{id}}$ is invertible. We proceed by proving the upper bound on $\|\mathbf{R}_{\mathsf{id}}\|_\infty$. First, observe

$\|\mathbf{R}'_i\|_\infty = \|\mathbf{R}_i\mathbf{G}^{-1}(\mathbf{H}_i G)\|_\infty \leq m$ since $\mathbf{R}_i, \mathbf{G}^{-1}(\mathbf{H}_i\mathbf{G}) \in \{-1,1\}^{m\times m}$ and therefore their product $\mathbf{R}'_i \in \mathbb{Z}_q^{m\times m}$ can not contain any element of absolute value larger than m. We then have

$$\|\mathbf{R}_{\mathsf{id}}\|_\infty = \left\|\mathbf{R} + \sum_{i=0}^{\ell} \mathbf{R}'_i\right\|_\infty \leq \|\mathbf{R}\|_\infty + \sum_{i=0}^{\ell} \|\mathbf{R}'_i\|_\infty \leq 1 + (\ell+1)m = \delta,$$

where the last inequality follows from $\mathbf{R} \in \{-1,1\}^{m\times m}$ and $\|\mathbf{R}'_i\|_\infty \leq m$. Finally, we have that for $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ it holds that for all $0 \leq i \leq \ell$ the distributions $(\mathbf{A}, \mathbf{A}')$ and $(\mathbf{A}, \mathbf{B}_i)$ have only negligible statistical difference by Lemma 7. The same holds for the distributions $(\mathbf{A}, \mathbf{A}')$ and $(\mathbf{A}, \mathbf{B})$.

□

3.4 Lattice-Based Identity-Based Key-Encapsulation-Mechanism

After we introduced our building blocks, we now describe our IB-KEM based on the IBE from CRYPTO'17 by Yamada [Yam17a]. To do so, we first formally define IB-KEMs and their security and the describe the construction.

Definition 5. An IB-KEM consists of the following four PPT algorithms:

- $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ takes as input the security parameter and outputs the public parameters mpk and the master secret key msk.
- $\mathsf{usk}_{\mathsf{id}} \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$ returns the user secret key $\mathsf{usk}_{\mathsf{id}}$ for identity $\mathsf{id} \in \{0,1\}^*$.
- $(\mathsf{ct}, K) \xleftarrow{\$} \mathsf{Encap}(\mathsf{mpk}, \mathsf{id})$ returns a tuple (ct, K), where ct is ciphertext encapsulating K with respect to identity id.
- $K = \mathsf{Decap}(\mathsf{usk}_{\mathsf{id}}, \mathsf{ct}, \mathsf{id})$ returns the decapsulated key K or an error symbol $\perp$.

For correctness we require that for all $\lambda \in \mathbb{N}$, all pairs $(\mathsf{mpk}, \mathsf{msk})$ generated by $\mathsf{Setup}(1^\lambda)$, all identities $\mathsf{id} \in \{0,1\}^*$, all (K, ct) output by $\mathsf{Encap}(\mathsf{mpk}, \mathsf{id})$ and all $\mathsf{usk}_{\mathsf{id}}$ generated by $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$:

$$\Pr[\mathsf{Decap}(\mathsf{usk}_{\mathsf{id}}, \mathsf{ct}, \mathsf{id}) = K] \geq 1 - \mathsf{negl}(\lambda).$$

We use the standard IND-CPA-security notion for IB-KEMs from [BFMS08].

Definition 6. Consider an adversary $\mathcal{A}$ with access (via oracle queries) to the procedures defined Figure 2. We say that $\mathcal{A}$ is *legitimate*, if $\mathcal{A}$ never queries $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id}^*)$, where id^* is the output of $\mathcal{A}_1$. We define the advantage of $\mathcal{A}$ in breaking the IND-ID-CPA security of IBKEM Π as

$$\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{A}}(\lambda) := \left|\Pr[\mathsf{IND\text{-}ID\text{-}CPA}^{\Pi}_{\mathcal{A}}(\lambda) = 1] - 1/2\right|$$

We include the running time of the security experiment into the running time $t_{\mathcal{A}}$ of $\mathcal{A}$. This will later allow us to simplify our security analysis and the statement of theorems.

$\underline{\mathsf{IND\text{-}ID\text{-}CPA}^{\Pi}_{\mathcal{A}}(\lambda)}$

$b \xleftarrow{\$} \{0,1\}$
$(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$(\mathsf{id}^*, st) \leftarrow \mathcal{A}_1^{\mathsf{KeyGen}(\mathsf{mpk},\mathsf{msk},\cdot)}(1^k, \mathsf{mpk})$
$K_0 \xleftarrow{\$} \mathcal{K}$; $(\mathsf{ct}, K_1) \xleftarrow{\$} \mathsf{Encap}(\mathsf{mpk}, \mathsf{id}^*)$
$b' \leftarrow \mathcal{A}_2^{\mathsf{KeyGen}(\mathsf{mpk},\mathsf{msk},\cdot)}(st, \mathsf{ct}, K_b)$
If $(b' == b)$ return 1, else 0

Figure 2: The security experiment for IB-KEMs, executed with scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ and adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracle $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$ returns $\mathsf{usk}_{\mathsf{id}} \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$ with the restriction that $\mathcal{A}$ is not allowed to query oracle $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$ for the target identity id^*.

The construction. Let $\mathcal{H} = \{H : \{0,1\}^* \to \{0,1\}^{2\lambda+3}\}$ be a family of hash functions, let $\ell = \lfloor \log(n) \rfloor$. Further, let $D_{\mathbb{Z}^m,\sigma}$ be the Gaussian distribution over $\mathbb{Z}^m$ with parameter $\sigma > 0$. Moreover, let $\mathsf{GenTrap}(1^n, 1^m, q)$ be an algorithm that outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ that is indistinguishable from a random matrix and a trapdoor $\mathbf{A}_{\sigma_0}^{-1}$ for $\sigma_0 = \omega(n \log q \log m)$. Note that for arbitrary $m' \geq m$, $\mathbf{u} \in \mathbb{Z}_q^n$ and $\mathbf{B} \in \mathbb{Z}_q^{n \times (m'-m)}$, the trapdoor $\mathbf{A}_{\sigma_0}^{-1}$ allows to sample vectors $\mathbf{v} \in \mathbb{Z}_q^{m'}$ from $D_{\mathbb{Z}^{m'},\sigma}$ conditioned on $[\mathbf{A} \mid \mathbf{B}]\mathbf{v} = \mathbf{u}$ for $\sigma' > \sigma_0$. We denote this as sampling from $[\mathbf{A} \mid \mathbf{B}]_\sigma^{-1}(\mathbf{u})$ as formalized in Section 3.1.

We now construct our IB-KEM scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ similar to [Yam17a] and based on LWE as follows.

Setup. $\mathsf{Setup}(1^\lambda)$ chooses parameters $n, m, q, \ell, \sigma, \alpha$ and α' as specified in Remark 1, where q is a prime. It runs $(\mathbf{A}, \mathbf{A}_{\sigma_0}^{-1}) \xleftarrow{\$} \mathsf{GenTrap}(1^n, 1^m, q)$ such that $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\sigma_0 = \omega(\sqrt{n \log(q) \log(m)})$ and then samples $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$. Finally, it samples $H \xleftarrow{\$} \mathcal{H}$ and $\mathbf{B}, (\mathbf{B}_i)_{0,\leq i \leq \ell}, \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{m \times m}$ and then outputs

$$\mathsf{mpk} = (H, \mathbf{A}, \mathbf{B}, (\mathbf{B}_i)_{0 \leq i \leq \ell}, \mathbf{C}, \mathbf{u}) \text{and} \qquad \mathsf{msk} := \mathbf{A}_{\sigma_0}^{-1}.$$

Key Generation. The algorithm KeyGen receives $(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$ as input and then computes the matrix $\mathbf{B}_{\mathsf{id}} := \mathsf{PubEval}(H, \mathsf{id}, \mathbf{B}, (\mathbf{B}_i)_{0 \leq i \leq \ell})$ such that $\mathbf{B} \in \mathbb{Z}_q^{m \times m}$. It then computes $[\mathbf{A} \mid \mathbf{C} + \mathbf{B}_{\mathsf{id}}]_\sigma^{-1}$ from $\mathbf{A}_{\sigma_0}^{-1}$ and samples $\mathbf{e} \xleftarrow{\$} [\mathbf{A} \mid \mathbf{C} + \mathbf{B}_{\mathsf{id}}]_\sigma^{-1}(\mathbf{u})$. It then outputs $\mathsf{usk}_{\mathsf{id}} := \mathbf{e} \in \mathbb{Z}^{2m}$.

Encapsulation. The Encap algorithm receives an identity $\mathsf{id} \in \{0,1\}^*$ and mpk as input. It computes $\mathbf{B}_{\mathsf{id}} := \mathsf{PubEval}(H, \mathsf{id}, \mathbf{B}, (\mathbf{B}_i)_{0 \leq i \leq \ell})$ such that $\mathbf{B}_{\mathsf{id}} \in \mathbb{Z}_q^{m \times m}$. It then samples $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, x_0 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha q}, \mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha' q}$ and $K \xleftarrow{\$} \{0,1\}$ and computes

$$c_0 = \mathbf{s}^\mathsf{T}\mathbf{u} + x_0 + K \cdot \lceil q/2 \rceil \in \mathbb{Z}_q, \qquad \mathbf{c}_1^\mathsf{T} = \mathbf{s}^\mathsf{T}[\mathbf{A} \mid \mathbf{C} + \mathbf{B}_{\mathsf{id}}] + [\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}] \in \mathbb{Z}_q^{2m}.$$

It then returns $(\mathsf{ct} = (c_0, \mathbf{c}_1), K)$.

Decapsulation. In order to decapsulate a ciphertext $\mathsf{ct} = (c_0, \mathbf{c}_1)$, the algorithm Decap receives the user secret key $\mathsf{usk}_{\mathsf{id}} = \mathbf{e}$ and computes $w = c_0 - \mathbf{c}_1^\mathsf{T} \cdot \mathbf{e} \in \mathbb{Z}_q$. It then returns $K := 1$ if $|w - \lceil q/2 \rceil| < \lceil q/4 \rceil$ and $K := 0$ otherwise.

Error term. We deduce the error term as Yamada in [Yam17b]. We have

$$w = c_0 - \mathbf{c}_1^\mathsf{T} \cdot \mathbf{e} = K \cdot \lceil q/2 \rfloor + x_0 - \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}\right] \cdot \mathbf{e},$$

where $x_0 - \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}\right] \cdot \mathbf{e}$ is the error term. Assuming $\alpha' \geq \alpha$, the error term is then bounded as follows

$$\begin{aligned}
\left| x_0 - \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}\right] \mathbf{e} \right| &\leq |x_0| + \left| \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}\right] \cdot \mathbf{e} \right| \\
&\leq |x_0| + \left\| \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}\right] \right\|_2 \cdot \|\mathbf{e}\|_2 \\
&\leq \alpha q \sqrt{m} + (\alpha' \sqrt{2m}) \cdot \sigma \sqrt{2m} \\
&= \mathcal{O}(\alpha' \sigma m q)
\end{aligned}$$

with overwhelming probability, where the first inequality follows from the triangle inequality, the second one follows from the Cauchy-Schwartz inequality, and the third follows from Lemma 4 Item 4 and Lemma 2. This implies the correctness of the scheme.

Remark 1. We select the parameters as described by Yamada (only in the full version [Yam17b]) with the additional constraint of n to be large enough to allow for blockwise partitioning. That is, we require

- that n' as chosen in Lemma 1 is at most n, that is $n \geq 2\lambda + 3$ as explained in Section 2,
- $\ell = \lfloor \log(n) \rfloor$ in order to use blockwise partitioning.
- the error is term is less than $q/5$ with overwhelming probability, that is $q > \Omega(\alpha' \sigma m q)$,
- that $\mathsf{GenTrap}$ can operate, that is $m > 6n \lceil \log q \rceil$,
- that the leftover hash lemma (Lemma 7) can be applied, meaning $m \geq (n+1)\log(q) + \omega(\log(n))$,
- σ has to be large enough such that the distribution of private keys in the actual scheme and in the reduction is the same, that is $\sigma > \sigma_0 = \omega(\sqrt{n \log(q) \log(m)})$ and $\sigma > m(1+\delta)\omega(\sqrt{\log(m)})$,
- that the ReRand algorithm can operate in the reduction, that is $\alpha'/2\alpha > \sqrt{2} \cdot m(\delta + 1)$ and $\alpha q > \omega(\sqrt{\log(m)})$ by Lemma 3.
- that the worst to average case reduction works, that is $\alpha q > 2\sqrt{2n}$.

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

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

Note that our compatible algorithms have $\delta = 1 + (\ell + 1)m$ compared to $\delta' = m^3 \mathcal{O}(\log^2(\lambda))(\mathcal{O}(\lambda) + 1)$ for Yamada's compatible algorithms for the modified admissible hash function and $\delta'' = \mathsf{poly}(\lambda)$ for his partitioning based on affine functions. This allows us to use much smaller q and σ.

3.5 Security of the IB-KEM

Our construction is secure when used in conjunction with the compatible algorithms we describe below in Section 3.3 under the $\mathsf{dLWE}_{n,m+1,q,\alpha}$ assumption.

Theorem 1. *If* $\Pi := (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ *from above is instantiated with a family* $\mathcal{H}$ *of weak near-collision resistant hash functions in the sense of Definition 1, then for any legitimate attacker* $\mathcal{A}$ *that breaks the* IND-ID-CPA *security of* Π *in time* $t_{\mathcal{A}}$ *with advantage* $\varepsilon_{\mathcal{A}} := \mathsf{Adv}^{\Pi}_{\mathcal{A}}(\lambda)$*, there exists an algorithm* $\mathcal{B}$ *that, given (sufficiently close approximations of)* $t_{\mathcal{A}}$ *and* $\varepsilon_{\mathcal{A}}$*, breaks the* $\mathsf{dLWE}_{n,m+1,q,\alpha}$ *assumption in time* $t_{\mathcal{B}} \approx t_{\mathcal{A}}$ *and with*

$$\mathsf{Adv}^{\mathsf{dLWE}_{n,m+1,q,\alpha}}_{\mathcal{B}}(\lambda) \geq \varepsilon^2_{\mathcal{A}}/(32t^2_{\mathcal{A}} - 16t_{\mathcal{A}}) - \mathsf{negl}(\lambda),$$

for some negligible term negl.

The proof of Theorem 1 mostly follows the proof from [Yam17b]. We provide it here for completeness.

PROOF. We prove Theorem 1 with a sequence of games argument [Sho04]. We denote with G_i the event that Game i outputs 1 and with $E_i := \Pr\left[1 \xleftarrow{\$} G_i\right] - 1/2$ the advantage of $\mathcal{A}$ in Game i. The first half of the proof closely follows the proofs by Jager, Kurek and Niehues [JK18, JN19]. The second half follows the proof by Yamada [Yam17b] (fullversion).

Game 0. This is the original IND-ID-CPA security experiment. We therefore have

$$\Pr[E_0] = \Pr\left[\mathsf{IND\text{-}ID\text{-}CPA}^{\mathcal{Q},\mathcal{A}}_{\Pi}(\lambda) = 1\right].$$

Game 1. This game is identical to Game 0, except that the challenger runs $\bar{K} = (K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(t_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$ from Lemma 1. Furthermore, it defines $\mathcal{I} := \{i : K_i \neq \perp\}$. Let $\mathcal{Q}$ be the set of all queries that the adversary makes to $\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \cdot)$, and let $\mathcal{Q}^* := \mathcal{Q} \cup \{\mathsf{id}^*\}$, where id^* is the challenge query. Additionally, the challenger raises event coll, aborts and outputs a random bit if there exist $\mathsf{id}, \mathsf{id}' \in \mathcal{Q}$ such that $\mathsf{id} \neq \mathsf{id}'$, but $H_i(\mathsf{id}) = H_i(\mathsf{id}')$ for all $i \in \mathcal{I}$. Since coll is defined exactly as in Lemma 1 we have

$$E_1 \geq E_0 - \Pr[\mathsf{coll}] = \varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}].$$

Game 2. In this game, the challenger raises event badChal, which occurs if there exists an index $i \in \mathcal{I}$ such that $H_i(\mathsf{id}^*) \neq K_i$ and it raises event badEval if there exists $\mathsf{id} \in \mathcal{Q}$ such that $H_i(\mathsf{id}) = K_i$ for all $i \in \mathcal{I}$. If either badChal or badEval occur, it aborts and outputs a random bit. By Property 2 of Lemma 1 we have $\mathsf{badEval} \implies \mathsf{coll} \vee \mathsf{badChal}$ and by Property 1 we have

$$E_2 = E_1 \cdot \Pr[\neg\mathsf{badChal}] = (\varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}]) \cdot \Pr[\neg\mathsf{badChal}] \geq \frac{\varepsilon^2_{\mathcal{A}}}{32t^2_{\mathcal{A}} - 16t_{\mathcal{A}}}.$$

From here on the proof closely follows Yamada's proof [Yam17b].

Game 3. In this game we change the way $\mathbf{B}, (\mathbf{B}_i)_{0\le i\le \ell}$ and $\mathbf{C}$ are chosen. The challenger computes $((\mathbf{B},\mathbf{R}),(\mathbf{B}_i,\mathbf{R}_i)_{0\le i\le\ell}) \xleftarrow{\$} \mathsf{Encode}(\mathbf{A},(K_i)_{0\le i\le\ell})$. The challenger then samples $\mathbf{R}_C \xleftarrow{\$} \{-1,1\}^{m\times m}$ and sets $\mathbf{C} := [\mathbf{A} \mid \mathbf{A}\mathbf{R}_C]$. Everything else remains as in Game 2. By Lemma 7 and the properties of Encode guaranteed by Definition 3, we have that the distributions

$$(\mathbf{A},\mathbf{C},\mathbf{B},(\mathbf{B}_i)_{0\le i\le\ell}) \qquad \text{and} \qquad (\mathbf{A},\mathbf{C}',\mathbf{B}',(\mathbf{B}'_i)_{0\le i\le\ell})$$

only have negligible statistical difference, where $\mathbf{C}',\mathbf{B}',(\mathbf{B}'_i)_{0\le i\le\ell} \xleftarrow{\$} \mathbb{Z}_q^{m\times m}$. We therefore have

$$E_3 \ge E_2 - \mathsf{negl}(\lambda).$$

We define $\mathsf{R}_{\mathsf{id}} := \mathsf{TrapEval}(H,\mathsf{id},\mathbf{R},(\mathbf{R}_i)_{0\le i\le\ell})$ before proceeding to the next game. Note that we have

$$\|\mathbf{R}_C + \mathbf{R}_{\mathsf{id}}\|_\infty \le \|\mathbf{R}_C\|_\infty + \|\mathbf{R}_{\mathsf{id}}\|_\infty \le 1+\delta, \tag{5}$$

where the last inequality follows from $\mathbf{R}_C \in \{-1,1\}^{m\times m}$ and that Encode is δ compatible with blockwise partitioning as specified in Definition 3. It also follows that

$$\mathbf{C} + \mathbf{B}_{\mathsf{id}} = \begin{cases} \mathbf{A}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}}) & \text{if } K_i = H_i(\mathsf{id}) \text{ for all } i\in\mathcal{I} \\ \mathbf{A}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}}) + \mathbf{H}_{\mathsf{id}}\mathbf{G} & \text{otherwise,} \end{cases} \tag{6}$$

where $\mathbf{H}_{\mathsf{id}} \in \mathbb{Z}_q^{n\times n}$ is invertible and $\mathbf{B}_{\mathsf{id}} = \mathsf{PubEval}(H,\mathsf{id},\mathbf{B},(\mathbf{B}_i)_{0\le i\le\ell})$.

Game 4. In this game, we change the way the $\mathbf{A}$ is sampled. Now, the challenger picks $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ instead of generating it with a trapdoor. By Lemma 4 Item 1, this makes only a negligible difference. For each key extraction query id the challenger now checks whether $H_i(\mathsf{id}) = K_i$ for all $i\in\mathcal{I}$ and if so outputs a random bit. By the second property in Lemma 1, this does not affect the view of the adversary. Otherwise, the challenger computes $[\mathbf{A} \mid \mathbf{C}+\mathbf{B}_{\mathsf{id}}]_\sigma^{-1} = [\mathbf{A} \mid \mathbf{A}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}} + \mathbf{H}_{\mathsf{id}}\mathbf{G})]_\sigma^{-1}$ from $\mathbf{R}_C + \mathbf{R}_{\mathsf{id}}$ for some invertible matrix $\mathbf{H}_{\mathsf{id}} \in \mathbb{Z}_q^{n\times n}$ that is guaranteed to exist since $(\mathsf{Encode},\mathsf{PubEval},\mathsf{TrapEval})$ fulfill the requirements of Definition 3. Observe that $\mathbf{H}_{\mathsf{id}}$ being invertible allows the reduction to answer the key extraction query using the "gadget"-trapdoor instead of a trapdoor for $\mathbf{A}$. The challenger then samples

$$\mathbf{e} \xleftarrow{\$} [\mathbf{A} \mid \mathbf{A}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}}) + \mathbf{H}_{\mathsf{id}}\mathbf{G}]_\sigma^{-1}(\mathbf{u}),$$

and returns it to $\mathcal{A}$. Note that since by Equation (5), the fact that $\mathbf{H}_{\mathsf{id}}$ is invertible and $\sigma > m(\delta+1)\cdot\omega(\sqrt{\log(m)})$, it follows from Lemma 4 and Lemma 6 that the changes above alter the view of the adversary only negligibly. Thus, we have that

$$E_4 \ge E_3 - \mathsf{negl}(\lambda)$$

Game 5. In this game, we change the way the challenge ciphertext for the challenge identity id^* is created. As in the previous games, the challenger checks whether $K_i = H_i(\mathsf{id}^*)$ holds for all $i\in\mathcal{I}$ and if so outputs a uniformly random bit and aborts. Otherwise, the challenger samples $b,\kappa_0,\kappa_1 \xleftarrow{\$} \{0,1\}, \mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, x_0 \xleftarrow{\$} D_{\mathbb{Z},\alpha q}, \bar{\mathbf{x}}_1 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha q}$ and sets $w_0 := \mathbf{s}^\mathsf{T}\mathbf{u} + x_0$ and $\mathbf{w}_1^\mathsf{T} := \mathbf{s}^\mathsf{T}\mathbf{A} + \bar{x}_1^\mathsf{T}$. Then, it computes $\mathbf{R}_{\mathsf{id}^*}$ using $\mathsf{TrapEval}$ as before and sets the challenge ciphertext $(c_0,\mathbf{c}_1) \in \mathbb{Z}_q \times \mathbb{Z}_q^{2m}$ as

$$c_0 := w_0 + \kappa_1\cdot\lceil q/2\rceil, \qquad \mathbf{c}_1^\mathsf{T} := \mathsf{ReRand}(\mathbf{w}_1,[\mathbf{I}_m \mid \mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*}],\alpha q,\alpha'/2\alpha), \tag{7}$$

where $\mathbf{I}_m$ is the identity matrix with size m. It then outputs $((c_0, \mathbf{c}_T), \kappa_b)$. Observe that this alters the view of the adversary only negligibly by Lemma 3. More precisely, set $\mathbf{V} := [\mathbf{I}_m \mid \mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*}], \mathbf{x} := \bar{\mathbf{x}}_1$ and $\mathbf{b}^\mathsf{T} := \mathbf{s}^\mathsf{T}\mathbf{A}$. Invoking Lemma 3 then yields that the distribution of $\mathbf{c}_1^\mathsf{T}$ as computed in Equation (7) above has only a negligible statistical distance to

$$\begin{aligned}\mathbf{c}_1^\mathsf{T} &= \mathbf{s}^\mathsf{T}\mathbf{A}^\mathsf{T}[\mathbf{I}_m \mid \mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*}] + [\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}] \\ &= \mathbf{s}^\mathsf{T}[\mathbf{A} \mid \mathbf{A}\mathbf{R}_C + \mathbf{A}\mathbf{R}_{\mathsf{id}^*}] + [\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}] \\ &= \mathbf{s}^\mathsf{T}[\mathbf{A} \mid \mathbf{C} + \mathbf{B}_{\mathsf{id}^*}] + [\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}],\end{aligned}$$

where $\mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} D_{\mathbb{Z}^m, \alpha' q}$. Note that the first equality holds because of Equation (6) and the last one because $\mathbf{C} = \mathbf{A}\mathbf{R}_C$ and $\mathbf{B}_{\mathsf{id}^*} = \mathbf{A}\mathbf{R}_{\mathsf{id}^*}$. We can apply Lemma 3 because

$$\alpha'/(2\alpha) > \sqrt{2} \cdot m \cdot (\delta + 1) \geq \sqrt{2m}\sqrt{m} \cdot \|\mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*}\|_\infty \geq \|\mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*}\|_2 ,$$

where the second inequality follows from Equation (5) with $\mathsf{id} = \mathsf{id}^*$ and the third from the general inequality between the two norms. We therefore conclude $E_5 \geq E_4 - \mathsf{negl}(\lambda)$.

Game 6. In this game, we further change how the challenge ciphertext is created. The 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 $\mathbf{R}_{\mathsf{id}^*}$ using $\mathsf{TrapEval}$ and samples $b, \kappa_0, \kappa_1 \xleftarrow{\$} \{0,1\}$. It then sets the challenge ciphertext as in Equation (7) but with $w_0 := v_0 + x_0$ and $\mathbf{w}_1 := \mathbf{v}_1 + \bar{\mathbf{x}}_1$. We claim that $E_6 \geq E_5 - \mathsf{negl}(\lambda)$ assuming $\mathsf{dLWE}_{n,m+1,q,\alpha}$. We show this by constructing an adversary $\mathcal{B}$ against $\mathsf{dLWE}_{n,m+1,q,\alpha}$ from an algorithm $\mathcal{A}$ distinguishing between Game 5 and Game 6 with a non-negligible advantage.

Given a $\mathsf{dLWE}_{n,m+1,q,\alpha}$ instance $(\mathbf{A}', \mathbf{w}' = \mathbf{v}' + \bar{x}) \in \mathbb{Z}_q^{n \times (m+1)} \times \mathbb{Z}_q^{m+1}$, where $\bar{\mathbf{x}} \xleftarrow{\$} D_{\mathbb{Z}^{m+1}, \alpha q}$. $\mathcal{B}$'s objective is to distinguish whether $\mathbf{v}'^\mathsf{T} = \mathbf{s}^\mathsf{T}\mathbf{A}'$ 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 let the last m columns of 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 $\mathbf{w}_1$. $\mathcal{B}$ then sets mpk as specified in Game 3. At any point in time, $\mathcal{B}$ aborts and outputs a random bit if $\mathcal{A}$ makes a key extraction query for an identity id with $K_i = H_i(\mathsf{id})$ for all $i \in \mathcal{I}$ or if there exists an $i \in \mathcal{I}$ such that $K_i \neq H_i(\mathsf{id}^*)$ for the challenge identity id^* submitted by $\mathcal{A}$. Since $\mathcal{B}$ set up mpk as described in Game 3 it can answer key extraction queries as described in Game 4 and in particular without knowledge of a trapdoor for $\mathbf{A}$. In order to answer $\mathcal{A}$'s challenge, $\mathcal{B}$ samples $b, \kappa_0, \kappa_1 \xleftarrow{\$} \{0,1\}$ and generates the challenge ciphertext $(c_0, \mathbf{c}_1^\mathsf{T})$ as in Equation (7) and outputs $((c_0, \mathbf{c}_1^\mathsf{T}), \kappa_b)$ to $\mathcal{A}$. Note that the secret randomness used to generate w_0 and $\mathbf{w}_1$ (namely, $\mathbf{s}$ and $\bar{\mathbf{x}}$) is not necessary for this computation. When $\mathcal{A}$ outputs its guess b', $\mathcal{B}$ checks whether $b' = b$ and if so outputs 1 and 0 otherwise.

One can easily verify that if $(\mathbf{A}', \mathbf{w}')$ is a valid LWE sample with $\mathbf{v}'^\mathsf{T} = \mathbf{s}^\mathsf{T}\mathbf{A}'$, 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 $\left(\Pr\left[1 \xleftarrow{\$} G_6\right] - \Pr\left[1 \xleftarrow{\$} G_5\right]\right)$. We thus have $E_6 \geq E_5 - \mathsf{Adv}_{\mathcal{B}}^{\mathsf{dLWE}_{n,m+1,q,\alpha}}(\lambda)$.

Game 7. We further randomize the challenge ciphertext in this game. The challenger now samples $v_0 \xleftarrow{\$} \mathbb{Z}_q, \mathbf{v}_1 \xleftarrow{\$} \mathbb{Z}_q^m, x_0 \xleftarrow{\$} D_{\mathbb{Z}, \alpha q}$ and $\mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} D_{\mathbb{Z}^m, \alpha' q}$ and computes $\mathsf{R}_{\mathsf{id}^*}$ using $\mathsf{TrapEval}$. Then the challenger samples $b, \kappa_0, \kappa_1 \xleftarrow{\$} \{0,1\}$ and sets the challenge ciphertext as

$$c_0 := v_0 + \kappa_1 \cdot \lceil q/2 \rceil \qquad \mathbf{c}_1^\mathsf{T} := \left[\mathbf{v}_1^\mathsf{T} \mid \mathbf{v}_1^\mathsf{T}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*})\right] + \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T}\right].$$

Similarly to the change from Game 4 to Game 5, by setting $\mathbf{V} := [\mathbf{I}_m \mid \mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*}], \mathbf{x} := \bar{\mathbf{x}}_1$ and $\mathbf{b} := \mathbf{v}_1$ (as in Game 6) and applying Lemma 3, it can be seen that this alters the view of the adversary only negligibly and we therefore have $E_7 \geq E_6 - \mathsf{negl}(\lambda)$.

Game 8. In this game, we change the challenge ciphertext to be a random vector in $\mathbb{Z}_q \times \mathbb{Z}_q^{2m}$ such that it information theoretically contains no information about κ_1 anymore. We therefore have $E_8 = 0$. We hence proceed to show that $E_8 \geq E_7 - \mathsf{negl}(\lambda)$. Since both games only differ in the generation of the challenge ciphertext, we focus on this part. First, observe that c_0 is uniformly random in both games since we already have $v_0 \xleftarrow{\$} \mathbb{Z}_q$ in Game 7. It remains to show that $\mathbf{c}_1 = \left[\mathbf{v}_1^\mathsf{T} \mid \mathbf{v}_1^\mathsf{T}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*})\right] + \left[\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2\mathsf{T}\right]$ is negligibly close to the uniform distribution over $\mathbb{Z}_q^{2m}$. First, observe that the following distributions are negligibly close:

$$\left(\mathbf{A}, \mathbf{A}\mathbf{R}_C, \mathbf{v}_1^\mathsf{T}, \mathbf{v}_1^\mathsf{T}\mathbf{R}_C\right) \approx \left(\mathbf{A}, \mathbf{A}', \mathbf{v}_1^\mathsf{T}, \mathbf{v}_1'^\mathsf{T}\right) \approx \left(\mathbf{A}, \mathbf{A}\mathbf{R}_C, \mathbf{v}_1^\mathsf{T}, \mathbf{v}_1'^\mathsf{T}\right), \tag{8}$$

here $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n\times m}, \mathbf{R}_C \xleftarrow{\$} \{-1,1\}^{m\times m}, \mathbf{v}_1, \mathbf{v}_1'^\mathsf{T} \xleftarrow{\$} \mathbb{Z}_q^m$. The first two distributions are negligibly close by Lemma 7 for $\left[\mathbf{A}^\mathsf{T} \mid \mathbf{v}_1\right]^\mathsf{T} \in \mathbb{Z}_q^{(n+1)\times m}$ and $\mathbf{R}_C$. We can see that the second and the third distribution are negligibly close by applying Lemma 7 for $\mathbf{A}$ and $\mathbf{R}_C$. Using this insight, we deduce that also the distributions

$$\begin{aligned}\left(\mathbf{A}, \mathbf{A}\mathbf{R}_C, \mathbf{c}_1^\mathsf{T}, \mathbf{v}_1'^\mathsf{T}\right) &\approx \left(\mathbf{A}, \mathbf{A}\mathbf{R}_C, \mathbf{v}_1^\mathsf{T} + \mathbf{x}_1^\mathsf{T}, \mathbf{v}_1'^\mathsf{T} + \mathbf{v}_1^\mathsf{T}\mathbf{R}_{\mathsf{id}^*} + \mathbf{x}_2^\mathsf{T}\right)\\ &\approx \left(\mathbf{A}, \mathbf{A}\mathbf{R}_C, \mathbf{v}_1^\mathsf{T} + \mathbf{x}_1^\mathsf{T}, \mathbf{v}_1^\mathsf{T}(\mathbf{R}_C + \mathbf{R}_{\mathsf{id}^*} + \mathbf{x}_2^\mathsf{T})\right)\end{aligned}$$

are negligibly close, where $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n\times m}, \mathbf{R}_C \xleftarrow{\$} \{-1,1\}^{m\times m}, \mathbf{v}_1, \mathbf{v}_1'^\mathsf{T} \xleftarrow{\$} \mathbb{Z}_q^m$. The first and second distribution are negligibly close because

1. $\mathbf{x}_1, \mathbf{x}_2$ are chosen independently at random from the other variables and
2. $\mathbf{R}_{\mathsf{id}^*}$ is computed from the trapdoors $(\mathbf{R}_i)_{0\leq i\leq \ell}$, which are also chosen independently at random from the other variables.

The second and third distributions are negligibly close by Equation (8). This yields $E_8 \geq E_7 - \mathsf{negl}(\lambda)$.

Analysis. From the above, it follows that

$$\begin{aligned}0 = E_8 &\geq E_6 - \mathsf{negl}(\lambda) \geq E_5 - \mathsf{Adv}_{\mathcal{B}}^{\mathsf{dLWE}_{n,m+1,q,\alpha}}(\lambda) - \mathsf{negl}(\lambda)\\ &\geq E_2 - \mathsf{Adv}_{\mathcal{B}}^{\mathsf{dLWE}_{n,m+1,q,\alpha}}(\lambda) - \mathsf{negl}(\lambda)\\ &\geq \frac{\varepsilon_{\mathcal{A}}^2}{32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}} - \mathsf{Adv}_{\mathcal{B}}^{\mathsf{dLWE}_{n,m+1,q,\alpha}}(\lambda) - \mathsf{negl}(\lambda).\end{aligned}$$

This is equivalent to

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{dLWE}_{n,m+1,q,\alpha}}(\lambda) \geq \frac{\varepsilon_{\mathcal{A}}^2}{32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}} - \mathsf{negl}(\lambda),$$

which concludes the proof of Theorem 1. □

4 IB-KEMs from Pairings

In this section, we show how to use blockwise partitioning to create two variants of the IB-KEMs of Boneh and Boyen [BB04a] and Waters [Wat05], respectively. In comparison to [BB04a], we achieve adaptive security instead of selective security. Additionally, we get ciphertexts of only a *single* element. In comparison to the corresponding construction from [JK18], the q of the required q-type assumption is reduced *quadratically*, while the tightness of the reduction is improved *quadratically*. In comparison to [Wat05], we have public parameters of size $O(\log \lambda)$ instead of $O(\lambda)$. The security analysis is also much simpler than in [Wat05] or the simplified proof by Bellare and Ristenpart [BR09]. To our best knowledge, this is the first adaptively-secure IBE scheme where ciphertexts consist only of *two* elements of a prime order algebraic group with logarithmic-size public parameters. For a better understanding we instantiate both constructions with symmetric pairings. However, asymmetric pairings work as well as it can be seen in [JK18].

Definition 7 (Definition 1 from [HJ16])**.** A *Bilinear Group Generator* is a probabilistic polynomial-time algorithm GrpGen that takes as input a security parameter 1^λ and outputs $\mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, \circ, \circ_T, e, \phi(1)) \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$ such that the following requirements are satisfied.

1. p is a prime and $\log(p) \in \Omega(\lambda)$
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
   a) $(\mathbb{G}, \circ)$ and $(\mathbb{G}_T, \circ_T)$ form algebraic groups,
   b) ϕ is a group isomorphism from $(\mathbb{Z}_p, +)$ to $(\mathbb{G}, \circ)$ and
   c) 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,

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

Encoding elements of $\{0,1\}^{2\lambda+3}$ as $\mathbb{Z}_p$-elements. Furthermore, in order to simplify the notation and description of the construction and its security analysis, we assume that elements of $\{0,1\}^{2\lambda+3}$ can be injectively encoded as elements of $\mathbb{Z}_p$.

4.1 Compact IB-KEM from Decisional Bilinear Diffie-Hellman

In this section we describe a variant of the IBE scheme of Waters [Wat05], which has public parameters of size $O(\log \lambda)$ instead of $O(\lambda)$.

The construction. Let $\mathcal{H} = \{H : \{0,1\}^* \to \{0,1\}^{2\lambda+3}\}$ be a family of hash functions, let $\ell = \lfloor \log(2\lambda + 3) \rfloor$, and let GrpGen be a bilinear group generator. We construct IB-KEM scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, allowbreak\mathsf{Encap}, \mathsf{Decap})$ as follows.

Setup. Choose a group description $\mathcal{BG} \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$, a random hash function $H \xleftarrow{\$} \mathcal{H}$, random generators $[1], [h] \in \mathbb{G}$, $\ell + 2$ random group elements $[u'], [u_0], \ldots, [u_\ell]$, and $x \xleftarrow{\$} \mathbb{Z}_p$. Compute $e([1], [hx]) = [hx]_T$ The master secret key is $msk := [hx]$. The public parameters are defined as

$$\mathsf{mpk} = ([1], [u'], [u_0], \ldots, [u_\ell], [hx]_T).$$

Key Generation. Let

$$[u(\mathsf{id})] := [u'] \prod_{i=0}^{\ell} [u_i]^{H_i(\mathsf{id})} = \left[u' + \sum_{i=0}^{\ell} u_i H_i(\mathsf{id}) \right]$$

To compute the private key for identity id, choose $s \xleftarrow{\$} \mathbb{Z}_p$ and compute and return

$$\mathsf{usk}_{\mathsf{id}} = ([s], [hx] \cdot [u(\mathsf{id})]^s = [hx + u(\mathsf{id})s])$$

Encapsulation. To encapsulate a key, choose $r \xleftarrow{\$} \mathbb{Z}_p$ and compute and return

$$\mathsf{ct} := ([r], [u(\mathsf{id})]^r = [u(\mathsf{id})r]) \in \mathbb{G}^2 \qquad \text{and} \qquad K := [hx]_T^r = [hxr]_T$$

Decapsulation. To recover K from a ciphertext $\mathsf{ct} = ([r], [u(\mathsf{id})r])$ and a matching user secret key $([s], [hx + u(\mathsf{id})s])$, compute and output

$$\frac{e([hx + u(\mathsf{id})s], [r])}{e([u(\mathsf{id})r], [s])} = \frac{[hxr + u(\mathsf{id})sr]_T}{[u(\mathsf{id})sr]_T} = [hxr]_T$$

Security Analysis. The security of this construction is based on the Decisional Bilinear Diffie-Hellman assumption, which is the same assumption as for schemes of [BB04a, Wat05]. In addition, we assume that the hash function H is weak near-collision resistant.

Definition 8 (Decisional Bilinear Diffie-Hellman [BB04a])**.** The advantage of an adversary $\mathcal{A}$ in solving the *Decisional Bilinear Diffie-Hellman Problem* (DBDH) with respect to a Bilinear Group Generator GrpGen is

$$\mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{A},\mathcal{BG}}(\lambda) := \left| \Pr\left[\mathcal{A}\left([\alpha], [\beta], [\gamma], V_0\right) = 1\right] - \Pr\left[\mathcal{A}\left([\alpha], [\beta], [\gamma], V_1\right) = 1\right] \right|,$$

where $\mathcal{BG} \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda), \alpha, \beta, \gamma \xleftarrow{\$} \mathbb{Z}_p, V_0 = [\alpha\beta\gamma]_T$ and $V_1 \xleftarrow{\$} \mathbb{G}_T$. We say that the DBDH assumption holds with respect to GrpGen, if $\mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{A}}(\lambda)$ is negligible for every PPT $\mathcal{A}$.

Theorem 2. *If Π is instantiated with a family $\mathcal{H}$ of weak near-collision resistant hash functions in the sense of Definition 1, then for any legitimate attacker $\mathcal{A}$ that breaks the* IND-ID-CPA *security of Π in time $t_\mathcal{A}$ with advantage $\varepsilon_\mathcal{A} := \mathsf{Adv}^{\Pi}_{\mathcal{A}}(\lambda)$, there exists an algorithm $\mathcal{B}$ that, given (sufficiently close approximations of) $t_\mathcal{A}$ and $\varepsilon_\mathcal{A}$, breaks the* DBDH *assumption in time $t_\mathcal{B} \approx t_\mathcal{A}$ and with*

$$\mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{B}}(\lambda) \geq \frac{\ell}{\ell + 1} \cdot \frac{\varepsilon^2_\mathcal{A}}{32t^2_\mathcal{A} - 16t_\mathcal{A}} - \mathsf{negl}(\lambda)$$

for some negligible term negl.

PROOF. Consider the following sequence of games, where we denote with G_i the event that Game i outputs 1 and with $E_i = \Pr\left[1 \xleftarrow{\$} G_i\right] - 1/2$ the advantage of $\mathcal{A}$ in Game i.

Game 0. This is the original $\mathsf{IND\text{-}ID\text{-}CPA}^{\Pi}_{\mathcal{A}}(\lambda)$ security experiment. By definition, we have

$$E_0 = \Pr[\mathsf{IND\text{-}ID\text{-}CPA}^{\Pi}_{\mathcal{A}}(\lambda) = 1] - 1/2 = \varepsilon_{\mathcal{A}}$$

Game 1. In this game, we additionally run algorithm

$$(K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(1^\lambda, t_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$$

at the beginning of the experiment, where algorithm BPSmp is from Lemma 1.

Furthermore, we define $\mathcal{I} := \{i : K_i \neq \perp\}$. Let $\mathcal{Q}$ be the set of all identities that the adversary queries to $\mathsf{KeyGen}(\mathsf{mpk}, msk, \cdot)$, and let $\mathcal{Q}^* := \mathcal{Q} \cup \{\mathsf{id}^*\}$, where id^* is the identity of the challenge ciphertext. We raise event coll, abort the experiment, and output a random bit, if there exists $i \in \mathcal{I}$ and $\mathsf{id}, \mathsf{id}' \in \mathcal{Q}^*$ such that $\mathsf{id} \neq \mathsf{id}'$, but $H_i(\mathsf{id}) = H_i(\mathsf{id}')$ for all $i \in \mathcal{I}$. Note that coll is defined exactly as in Lemma 1 and that we have

$$E_1 \geq E_0 - \Pr[\mathsf{coll}] = \varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}].$$

Game 2. We raise event $\mathsf{badChal}$, output a random bit, and abort the game, if there exist $i \in \mathcal{I}$ such that $K_i \neq H(\mathsf{id}^*)$. Note that $\mathsf{badChal}$ is defined exactly as in Lemma 1 and that we have

$$E_2 = E_1 \cdot \Pr[\neg\mathsf{badChal}] = (\varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}]) \cdot \Pr[\neg\mathsf{badChal}] \geq \varepsilon^2_{\mathcal{A}}/(32t^2_{\mathcal{A}} - 16t_{\mathcal{A}})$$

where the last inequality is from Property 1 of Lemma 1.

Game 3. This game deviates from the security proofs of other constructions in this paper. We need to deal with an event dlog, which is defined below, in order to apply blockwise partitioning to the Boneh-Boyen/Waters scheme.

First of all, we modify the way how the experiment samples the group elements that determine the function $[u(\mathsf{id})]$. The experiment first chooses a generator $[\alpha] \xleftarrow{\$} \mathbb{G}$ and $r', r_1, \ldots, r_\ell \xleftarrow{\$} \mathbb{Z}_p$ uniformly random. Then it sets

$$[u_i] := \begin{cases} [\alpha r_i] & \text{if } K_i \neq \perp \\ [r_i] & \text{if } K_i = \perp \end{cases} \qquad \text{and} \qquad [u'] := \left[r' - \sum_{i \in \mathcal{I}} \alpha r_i K_i \right] \tag{9}$$

Note that the distribution of these values is still uniform, and therefore identical to Game 2. Game 3 now raises event dlog and aborts, if there exists $\mathsf{id} \in \mathcal{Q}$ such that

$$\sum_{i \in \mathcal{I}} \alpha r_i K_i = \sum_{i \in \mathcal{I}} \alpha r_i H_i(\mathsf{id}) \iff \sum_{i \in \mathcal{I}} r_i (K_i - H_i(\mathsf{id})) = 0 \tag{10}$$

We claim that there exists an algorithm $\mathcal{B}_1$ that breaks the Decisional Bilinear Diffie-Hellman assumption with success probability $\Pr[\mathsf{dlog}]/|\mathcal{I}|$. $\mathcal{B}_1$ receives as input $(\mathcal{BG}, [r], [\beta], [\gamma], V)$. It will compute the discrete logarithm r and use this to break the DBDH assumption.[1]

$\mathcal{B}_1$ picks $j \xleftarrow{\$} \mathcal{I}$ at random and defines $[r_j] := [r]$. Then it proceeds exactly like Game 3. If dlog occurs, then Equation (10) holds. Due to Game 2 we can be certain that $K_i = H_i(\mathsf{id}^*)$ holds for all $i \in \mathcal{I}$, and due

[1] We could alternatively reduce to the weaker discrete logarithm problem, but we will later reduce to DBDH anyway, so omitting the additional definition saves trees.

to Game 1 we know that for any $\mathsf{id} \in \mathcal{Q}$ there must be at least one $i \in \mathcal{I}$ such that $H_i(\mathsf{id}) \neq H_i(\mathsf{id}^*)$. These two together yield that for any $\mathsf{id} \in \mathcal{Q}$ there must exist at least one $i \in \mathcal{I}$ such that $H_i(\mathsf{id}) \neq K_i$.

Let id be the first identity for which Equation (10) holds. With probability at least $1/|\mathcal{I}|$ we have $j = i$. In this case $\mathcal{B}_1$ is able to compute

$$r = r_j = \frac{\sum_{i \in \mathcal{I} \setminus \{j\}} r_i (K_i - H_i(\mathsf{id}))}{H_j(\mathsf{id}) - K_j}$$

which immediately enables $\mathcal{B}_1$ to test whether $V = e([\beta], [\gamma])^r$ holds. With $|\mathcal{I}| \leq \ell$ we thus get

$$E_3 \geq E_2 - \mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{B}_1}(\lambda)/\ell$$

Reduction. Now we are able to describe our final reduction $\mathcal{B}_2$ to the Decisional Bilinear Diffie-Hellman assumption. $\mathcal{B}_2$ that receives $(\mathcal{BG}, [\alpha], [\beta], [\gamma], V)$ and simulates the IND-ID-CPA experiment as follows.

Setup. $\mathcal{B}_2$ defines $[x] := [\alpha]$, $[h] := [\beta]$, and uses $[\alpha]$ to compute $[u'], [u_0], \ldots, [u_\ell]$ exactly as in Game 3, Equation (9). The master public parameters are defined as

$$\mathsf{mpk} = ([1], [u'], [u_0], \ldots, [u_\ell], e([\alpha], [\beta]))$$

note that this this is a correctly distributed master public key. The secret key is implicitly defined as $[\alpha\beta]$.

Key Generation. In the sequel let us write

$$a(\mathsf{id}) := \sum_{i \in \mathcal{I}} r_i (H_i(\mathsf{id}) - K_i) \qquad \text{and} \qquad b(\mathsf{id}) := r' + \sum_{i \notin \mathcal{I}} r_i H_i(\mathsf{id})$$

such that we have $[u(\mathsf{id})] = [\alpha a(\mathsf{id}) + b(\mathsf{id})]$.

$\mathcal{B}_2$ needs to compute a secret key of the form

$$[s], [\alpha\beta + u(\mathsf{id})s]$$

such that s is uniform over $\mathbb{Z}_p$. To this end, it picks $s' \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$[s] := [\beta]^{-1/a(\mathsf{id})} \cdot [s']$$

which is correctly distributed and implicitly defines $s := -\beta/a(\mathsf{id}) + s'$. Then it computes

$$\begin{aligned}
[z] &:= [\beta]^{-b(\mathsf{id})/a(\mathsf{id})} \cdot [\alpha]^{a(\mathsf{id})s'} \cdot [b(\mathsf{id})s'] \\
&= [\alpha\beta - \alpha\beta - \beta b(\mathsf{id})/a(\mathsf{id}) + \alpha a(\mathsf{id})s' + b(\mathsf{id})s'] \\
&= [\alpha\beta + (\alpha a(\mathsf{id}) + b(\mathsf{id}))(-\beta/a(\mathsf{id}) + s')] \\
&= [\alpha\beta + u(\mathsf{id})s]
\end{aligned}$$

Note here that we have $a(\mathsf{id}) \neq 0$ for all $\mathsf{id} \in \mathcal{Q}$, as otherwise we raise event dlog and abort due to Game 3, Equation (10). Then it returns $([s], [z])$.

Encapsulation. Given a challenge identity id^*, $\mathcal{B}_2$ has to create a challenge ciphertext of the form

$$\mathsf{ct} := ([r], [u(\mathsf{id}^*)r])$$

$\mathcal{B}_2$ sets $[r] := [\gamma]$, where $[\gamma]$ is from the DBDH challenge. Note that we have $a(\mathsf{id}^*) = 0$, as otherwise we raise event guess and abort due to Game 2, and thus

$$[u(\mathsf{id}^*)\gamma] = [b(\mathsf{id}^*)\gamma] = [\gamma]^{b(\mathsf{id}^*)}$$

such that $\mathsf{ct}^* = ([\gamma], [\gamma]^{b(\mathsf{id}^*)})$ is a consistent ciphertext.

Finally, it sets $K^* := T$ and returns (ct^*, K^*).

Note that if $T = [\alpha\beta\gamma]_T$, then this is a correct key, since for any valid user key $([s], [hx + u(\mathsf{id}^*)s])$ for the challenge identity id^* we have

$$\frac{e([\alpha\beta + u(\mathsf{id})s], [\gamma])}{e([u(\mathsf{id})\gamma], [s])} = \frac{[\alpha\beta\gamma + u(\mathsf{id})s\gamma]_T}{[u(\mathsf{id})s\gamma]_T} = [\alpha\beta\gamma]_T$$

while if T is random, then so is K^*. Hence, $\mathcal{B}_2$ provides a perfect simulation of Game 3. It returns whatever $\mathcal{A}$ returns, and thus we have that

$$\mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{B}_2}(\lambda) \geq E_3.$$

By collecting probability across all games, we get

$$\frac{\mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{B}_1}(\lambda)}{\ell} + \mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{B}_2}(\lambda) \geq \frac{\varepsilon^2_{\mathcal{A}}}{32t^2_{\mathcal{A}} - 16t_{\mathcal{A}}}.$$

□

4.2 IB-KEM with Short Ciphertexts

In this section, we present a new IB-KEM that is adaptively secure and where the ciphertext consists of only a *single* element. Compared to the only other construction with these properties ([JK18]), the q of the required q-type assumption is reduced *quadratically*, while the tightness of the reduction is improved *quadratically*, as well. Due to weak near-collision resistance, we are also able to reduce the output length of the hash function to roughly half of the output length required in [JK18], which reduces computational costs while guaranteeing the same level of security.

The construction. Let $\mathcal{H} = \{H : \{0,1\}^* \to \{0,1\}^{2\lambda+3}\}$ be a family of hash functions, let $\ell = \lfloor \log(2\lambda + 3) \rfloor$, and let GrpGen be a bilinear group generator. We construct the IB-KEM scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ as follows.

Setup. Choose a group description $\mathcal{BG} \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$, a random hash function $H \xleftarrow{\$} \mathcal{H}$, a random generator $[1] \in \mathbb{G}_1$, and random elements $x_0, \ldots, x_\ell \in \mathbb{Z}_p^*$. Define the master secret key msk as

$$\mathsf{msk} = (x_0, \ldots, x_\ell) \in \mathbb{Z}_p^{\ell+1}.$$

For $i \in \mathbb{N}$ and $m \in \mathbb{N}_0$ define $b_i(m)$ as the function that, on input of integer m, outputs the i-th bit of the binary representation of m. For $\mathsf{msk} = (x_0, \ldots, x_\ell)$ and $m = 0, \ldots, 2^{\ell+1} - 1$ define

$$F(\mathsf{msk}, m) := \prod_{i=0}^{\ell} x_i^{b_i(m)}. \tag{11}$$

The public parameters are defined as

$$\mathsf{mpk} = ([F(\mathsf{msk}, 0)], \ldots, [F(\mathsf{msk}, 2^{\ell+1} - 1]).$$

Key Generation. Let

$$u(\mathsf{id}) = \prod_{i=0}^{\ell} (H_i(\mathsf{id}) + x_i) \in \mathbb{Z}_p. \tag{12}$$

Then the private key for identity id is computed as $\mathsf{usk}_{\mathsf{id}} = [1/u(\mathsf{id})]$.

Encapsulation. Observe that

$$u(\mathsf{id}) = \prod_{i=0}^{\ell} (H_i(\mathsf{id}) + x_i) = d_0 + \sum_{m=1}^{2^\ell - 1} \left(d_m \prod_{i=0}^{\ell} x_i^{b_i(n)} \right),$$

where the constants d_i are efficiently computable from $H(\mathsf{id})$. Using $H(\mathsf{id})$ and mpk first $[u(\mathsf{id})]$ is computed as

$$[u(\mathsf{id})] = \left[d_0 + \sum_{m=1}^{2^\ell - 1} \left(d_m \prod_{i=0}^{\ell} x_i^{b_i(n)} \right) \right] = [d_0] \cdot \prod_{m=1}^{2^\ell - 1} [F(\mathsf{msk}, m)]^{d_m}.$$

Note that this does not require knowledge of $x_0, \ldots, x_\ell$ explicitly.

Finally, the ciphertext and key are computed as

$$(\mathsf{ct}, K) = ([u(\mathsf{id})]^r, e([1], [1])^r) \in \mathbb{G}_1 \times \mathbb{G}_T.$$

for a uniformly random $r \xleftarrow{\$} \mathbb{Z}_p$.

Decapsulation. To recover K from a ciphertext ct for identity id and a matching user secret key $[1/(u(\mathsf{id}))]$, compute and output $e(C, \mathsf{usk}_{\mathsf{id}})$.

Security Analysis. The security of this construction is based on the q-DBDHI assumption, which is the same assumption as for the scheme of [BB04a]. In addition, we assume that the hash function H is weak near-collision resistant.

Definition 9 (q-Decision Bilinear Diffie-Hellman Inversion Assumption [BB04a])**.** For a PPT algorithm $\mathcal{A}$, the advantage of $\mathcal{A}$ in solving the *q-Decision Bilinear Diffie-Hellman Inversion Problem* (q-DBDHI) with respect to a Bilinear Group Generator GrpGen is

$$\begin{aligned} \mathsf{Adv}_{\mathcal{A}}^{q\text{-DBDHI}}(\lambda) := \big| &\Pr\left[\mathcal{A}\left(\mathcal{BG}, [y], [y\alpha], [y\alpha^2], \ldots, [y\alpha^q], V_0\right) = 1\right] - \\ &\Pr\left[\mathcal{A}\left(\mathcal{BG}, [y], [y\alpha], [y\alpha^2], \ldots, [y\alpha^q], V_1\right) = 1\right] \big|, \end{aligned}$$

where $\mathcal{BG} \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda), \alpha \xleftarrow{\$} \mathbb{Z}_p^*, [y] \xleftarrow{\$} \mathbb{G}, V_0 = e([y],[y])^{1/\alpha}$ and $V_1 \xleftarrow{\$} \mathbb{G}_T$. The probability is over the randomness of $\mathcal{A}$, GrpGen and sampling $\alpha, \tilde{g}, h$ and V_1. We say that the q-DBDHI assumption holds with respect to GrpGen if $\mathsf{Adv}^{q\text{-}\mathsf{DBDHI}}_{\mathcal{A}}(\lambda)$ is negligible for every PPT $\mathcal{A}$.

We start by defining the strength of the q-DBDHI assumption. That is, we set

$$q := 4\lambda + 7 + j + 2 \sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} \left(2^{2^i} - 1\right).$$

Using the following lemma, we immediately obtain $q \leq 4\lambda + 8 + \lfloor \log(2\lambda+3) \rfloor + 32t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}}$ because $j \leq \lfloor \log(2\lambda+3) \rfloor + 1$.

Lemma 11. *Let $\mathcal{I} = \{i : K_i \neq \perp\}$ be as above, then*

$$2 \cdot \sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ i \in \mathcal{I}}} \left(2^{2^i} - 1\right) \leq \frac{32t_{\mathcal{A}}^2}{\varepsilon_{\mathcal{A}}}.$$

The proof of Lemma 11 consists only of simple arithmetic and we therefore provide it in Appendix B.

Theorem 3. *If Π is instantiated with a family $\mathcal{H}$ of weak near-collision resistant hash functions in the sense of Definition 1, then for any legitimate attacker $\mathcal{A}$ that breaks the* $\mathsf{IND\text{-}ID\text{-}CPA}$ *security of Π in time $t_{\mathcal{A}}$ with advantage $\varepsilon_{\mathcal{A}} := \mathsf{Adv}^{\Pi}_{\mathcal{A}}(\lambda)$, there exists an algorithm $\mathcal{B}$ that, given (sufficiently close approximations of) $t_{\mathcal{A}}$ and $\varepsilon_{\mathcal{A}}$, breaks the q-DBDHI assumption with $q \leq 4\lambda + 9 + \lfloor \log(2\lambda+3) \rfloor + 32t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}}$ in time $t_{\mathcal{B}} = O(32t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}})$ and with*

$$\mathsf{Adv}^{q\text{-}\mathsf{DBDHI}}_{\mathcal{B}}(\lambda) \geq \varepsilon_{\mathcal{A}}^2/(32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}) - \mathsf{negl}(\lambda),$$

for some negligible term negl.

PROOF. Consider the following sequence of games. We denote with G_i the event that Game i outputs 1 and with $E_i := \Pr\left[1 \xleftarrow{\$} G_i\right] - 1/2$ the advantage of $\mathcal{A}$ in Game i.

Game 0. This is the original $\mathsf{IND\text{-}ID\text{-}CPA}^{\Pi}_{\mathcal{A}}(\lambda)$ security experiment. By definition, we have

$$E_0 = \Pr[\mathsf{IND\text{-}ID\text{-}CPA}_{\mathcal{A}}\Pi(\lambda) = 1] - 1/2 = \varepsilon_{\mathcal{A}}.$$

Game 1. This game is identical to Game 0, except that the challenger runs $\bar{K} = (K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(t_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$ from Lemma 1. Furthermore, it defines $\mathcal{I} := \{i : K_i \neq \perp\}$. Let $\mathcal{Q}$ be the set of all queries that the adversary queries to $\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \cdot)$, and let $\mathcal{Q}^* := \mathcal{Q} \cup \{\mathsf{id}^*\}$, where id^* is the challenge query. Additionally, the challenger raises event coll, aborts and outputs a random bit if there exist $\mathsf{id}, \mathsf{id}' \in \mathcal{Q}$ such that $\mathsf{id} \neq \mathsf{id}'$, but $H_i(\mathsf{id}) = H_i(\mathsf{id}')$ for all $i \in \mathcal{I}$. Since coll is defined exactly as in Lemma 1 we have

$$E_1 \geq E_0 - \Pr[\mathsf{coll}] = \varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}]$$

Game 2. In this game, the challenger raises event $\mathsf{badChal}$ which occurs if there exists an index $i \in \mathcal{I}$ such that $H_i(\mathsf{id}^*) \neq K_i$ and it raises event $\mathsf{badEval}$ if there exists $\mathsf{id} \in \mathcal{Q}$ such that $H_i(\mathsf{id}) = K_i$ for all $i \in \mathcal{I}$. If either $\mathsf{badChal}$ or $\mathsf{badEval}$ occur it aborts and outputs a random bit. By Property 2 of Lemma 1 we have $\mathsf{badEval} \implies \mathsf{coll} \vee \mathsf{badChal}$ and by Property 1 we have

$$E_2 = E_1 \cdot \Pr[\neg\mathsf{badChal}] = (\varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}]) \cdot \Pr[\neg\mathsf{badChal}] \geq \frac{\varepsilon_{\mathcal{A}}^2}{32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}}$$

Game 3. In this game, we change the way msk and mpk are generated by the challenger. It samples $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$, $\tilde{x}_i \xleftarrow{\$} \mathbb{Z}_p^*$ and sets for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$

$$x_i := \begin{cases} \tilde{x}_i \cdot \alpha - K_i & \text{if } K_i \neq \perp \\ \tilde{x}_i & \text{otherwise.} \end{cases}$$

If $x_i = 0$ for any $0 \leq i \leq \lfloor \log(2\lambda + 3) \rfloor$, then the challenger aborts and outputs a random bit. This happens *iff* $\tilde{x}_i \alpha = K_i$. Since $\tilde{x}_i \alpha$ is distributed uniformly at random in $\mathbb{Z}_p^*$, the probability that this happens for any of the $\mathcal{O}(\log \lambda)$ many x_i is negligible and we therefore have $E_3 = E_2 - \mathsf{negl}(\lambda)$.

Game 4. In this game, the way the challenger chooses $[1]$ is changed. Let $j = |\mathcal{I}|$ be the number of non-wildcard positions in $\bar{K}$. The challenger then first defines the polynomial $\mathsf{Q}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ as

$$\mathsf{Q}(\mathsf{Z}) := \mathsf{Z}^{j-1} \prod_{\substack{i=0 \\ K_i \neq \perp}}^{\lfloor \log(2\lambda+3) \rfloor} \prod_{\substack{-2^{2^i}+1 \leq k \leq 2^{2^i}-1 \\ k \neq 0}} (\tilde{x}_i \mathsf{Z} + k)\,. \tag{13}$$

The challenger samples $[y] \xleftarrow{\$} \mathbb{G}_1$ and sets $[1] := [y]^{\mathsf{Q}(\alpha)}$. If $[1] = 1_{\mathbb{G}}$, which happens *iff* $\mathsf{Q}(\alpha) \equiv 0 \bmod p$, the challenger outputs a random bit and aborts. It can be seen that the distribution of $[1]$ changes only if $\mathsf{Q}(\alpha) \equiv 0 \bmod p$. By Lemma 11, we have $\deg(\mathsf{Q}) \leq \lfloor \log(2\lambda + 3) \rfloor + 8 + 32t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}}$. Since α is uniformly random in $\mathbb{Z}_p^*$, we have by the Schwartz-Zippel Lemma $\Pr[\mathsf{Q}(\alpha) = 0] \leq \frac{\deg(Q)}{p-1}$. Due to the fact that $\deg(\mathsf{Q})$ is polynomial in λ and $p \in 2^{\Omega(\lambda)}$ by the properties of GrpGen, we have $E_4 = E_3 - \mathsf{negl}(\lambda)$.

The previous steps now enable us to construct a an adversary $\mathcal{B}$ against the q-DBDHI that perfectly simulates Game 4. $\mathcal{B}$ operates as follows.

Initialization of $\mathcal{B}$. $\mathcal{B}$ runs $\bar{K} = (K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(t_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$ at the beginning of the experiment and by doing so it also determines the index set $\mathcal{I} := \{i : K_i \neq \perp\}$ according to Lemma 1. Moreover, it receives a q-DBDHI instance $(\mathcal{BG}, [y], [y\alpha], \ldots, [y\alpha^q], V)$, where $q = 4\lambda + 6 + j + 2\sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} \left(2^{2^i} - 1\right)$, where $j = |\mathcal{I}|$ is the number of non-wildcard positions in $\bar{K}$, and where either $V = e([y], [y]s)^{1/\alpha}$ or $V \xleftarrow{\$} \mathbb{G}_T$. In order to define $[1]$, $\mathcal{B}$ samples $\tilde{x}_i \xleftarrow{\$} \mathbb{Z}_p^*$ for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$. Then it computes coefficients $\varphi_0, \ldots \varphi_{q-4\lambda-8} \in \mathbb{Z}_p$ such that

$$\mathsf{Q}(\mathsf{Z}) = \mathsf{Z}^{j-1} \prod_{\substack{i=0 \\ K_i \neq \perp}}^{\lfloor \log(2\lambda+3) \rfloor} \prod_{\substack{-2^{2^i}+1 \leq k \leq 2^{2^i}-1 \\ k \neq 0}} (\tilde{x}_i \mathsf{Z} + k) = \sum_{k=0}^{q-4\lambda-8} \varphi_k \mathsf{Z}^k.$$

Then $\mathcal{B}$ defines

$$[1] := \prod_{k=0}^{q-8\lambda-8} [y\alpha^k]^{\varphi_k} = [y \sum_{k=0}^{q-4\lambda-8} \varphi_k \alpha^k] = [y\mathsf{Q}(\alpha].$$

If $[1] = 1_{\mathbb{G}}$, meaning $\mathsf{Q}(\alpha) \equiv 0 \bmod p$, $\mathcal{B}$ aborts and outputs a random bit. Then $\mathcal{B}$ proceeds with computing $[F(\mathsf{msk}, m)]$ for $m = 0, ..., 2^{\ell+1} - 1$, where $\ell = \lfloor \log(2\lambda + 3) \rfloor$ and $F(\mathsf{msk}, m) = \prod_{i'=0}^{\ell} x_{i'}^{b_{i'}(m)}$ as in Equation (11). First $\mathcal{B}$ samples $\tilde{x_{i'}}$ for $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$ and then it (implicitly) sets

$$x_{i'} := \begin{cases} \tilde{x}_{i'} \cdot \alpha - K_{i'} & \text{if } K_i \neq \perp \\ \tilde{x}_{i'} & \text{otherwise} \end{cases}.$$

Then it computes $\psi_{m,0}, \ldots, \psi_{m,q} \in \mathbb{Z}_p$ for all $m = 0, ..., 2^{\ell+1} - 1$ such that

$$\mathsf{Q}(\alpha) \cdot \prod_{i'=0}^{\ell} x_{i'}^{b_{i'}(m)} = \sum_{k=0}^{q} \psi_{m,k} \alpha^k.$$

Afterwards, it sets for all $m = 0, \ldots, 2^{\ell+1} - 1$

$$[F(\mathsf{msk}, m)] := \prod_{k=0}^{q} [y\alpha^k]^{\psi_{m,k}} = [y\mathsf{Q}(\alpha) \cdot \prod_{i'=1}^{\ell} x_{i'}^{b_{i'}(m)}].$$

Finally $\mathcal{B}$ outputs $\mathsf{mpk} = ([F(\mathsf{msk}, 0)], \ldots, [F(\mathsf{msk}, 2^{\ell+1} - 1], H)$ to $\mathcal{A}$.

In order to respond to KeyGen queries and the challenge query for id^*, $\mathcal{B}$ defines a polynomial $\mathsf{P}_{\mathsf{id}}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$, which will assume the role of $u(\mathsf{id})$. Let $x_{i'} = \tilde{x}_{i'} \cdot Z - K_{i'}$ if $K_i \neq \perp$ and $x_{i'} = \tilde{x}_{i'}$ else. Then

$$\mathsf{P}_{\mathsf{id}}(\mathsf{Z}) := \prod_{i'=1}^{\lfloor \log(2\lambda+3) \rfloor} (x_{i'} + H_{i'}(\mathsf{id})).$$

We require the following lemma by Yamada [Yam17a] to proceed with the description of $\mathcal{B}$.

Lemma 12 (Lemma 6 in [Yam17a])**.** *Let* $\mathsf{id} \in \{0,1\}^*$ *then there exist* $\zeta_{\mathsf{id}} \in \mathbb{Z}_p^*$ *and* $\mathsf{R}_{\mathsf{id}}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ *such that*

$$\frac{\mathsf{Q}(\mathsf{Z})}{\mathsf{P}_{\mathsf{id}}(\mathsf{Z})} = \begin{cases} \frac{\zeta_{\mathsf{id}}}{\mathsf{Z}} + \mathsf{R}_{\mathsf{id}}(\mathsf{Z}) & \textit{if } H_i(\mathsf{id}) = K_i \textit{ for all } i \in \mathcal{I} \\ \mathsf{R}_{\mathsf{id}}(\mathsf{Z}) & \textit{else} \end{cases}.$$

The proof of Lemma 12 is identical to the proof of Lemma 7 in [Yam17a] and we provide it in Section C for completeness.

Answering key queries. When $\mathcal{B}$ receives a key query for $\mathsf{id} \in \{0,1\}^*$ from $\mathcal{A}$, it checks whether $H_i(\mathsf{id}) = K_i$ for all $i \in \mathcal{I}$ and if so aborts and outputs a random bit. If there is an index $i \in \mathcal{I}$ such that $H_i(\mathsf{id}) \neq K_i$ let $\mathsf{R}_{\mathsf{id}}(\mathsf{Z}) = \sum_{k=0}^{q} \mathsf{Z}^k \rho_{\mathsf{id},k} \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{R}_{\mathsf{id}}(\mathsf{Z}) = \mathsf{Q}(\mathsf{Z})/\mathsf{P}_{\mathsf{id}}(\mathsf{Z})$, which is guaranteed to exist by Lemma 12. First $\mathcal{B}$ computes the coefficients $\rho_{\mathsf{id},k} \in \mathbb{Z}_p$ for all k. Then it computes and returns

$$[1/u(\mathsf{id})] := \prod_{k=0}^{q} [y\alpha^k]^{\rho_{\mathsf{id},k}} = [y \sum_{k=0}^{q} \alpha^k \rho_{\mathsf{id},k}] = [y\mathsf{R}_{\mathsf{id}}(\alpha)] = [y\mathsf{Q}(\alpha)/\mathsf{P}_{\mathsf{id}}(\alpha)].$$

Answering challenge. When $\mathcal{B}$ receives the challenge $\mathsf{id}^* \in \{0,1\}^*$, it checks whether there exists $i \in \mathcal{I}$ such that $H_i(\mathsf{id}) \neq K_i$. If this is true then it aborts and outputs a random bit. Else, let $\zeta := \zeta_{\mathsf{id}^*} \in \mathbb{Z}_p$ and $\mathsf{R}(\mathsf{Z}) := \mathsf{R}_{\mathsf{id}^*}(\mathsf{Z}) = \sum_{k=0}^{q} \mathsf{Z}^k \gamma_k \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{Q}(\mathsf{Z})/\mathsf{P}_{id^*}(\mathsf{Z}) = \zeta/\mathsf{Z} + \mathsf{R}(\mathsf{Z})$ as guaranteed by Lemma 12. $\mathcal{B}$ then computes coefficients $\gamma_k \in \mathbb{Z}_p$ of R.
Using that $\mathsf{Q}(\mathsf{Z}) = \sum_{k=0}^{q-4\lambda-8} \varphi_k \mathsf{Z}^k$ it also computes

$$\tilde{K} := V^{\zeta\cdot\varphi_0} e\left([y], \prod_{k=1}^{q-4\lambda-8} [y\alpha^{k-1}]^{\zeta\varphi_k}\right) e\left(\prod_{k=0}^{q} [y\alpha^k]^{\gamma_k}, \prod_{k=0}^{q-4\lambda-8} [y\alpha^k]^{\varphi_k}\right) \tag{14}$$

Afterwards it samples $L \xleftarrow{\$} \mathbb{Z}_p$ and computes and outputs

$$K := \tilde{K}^L \qquad \text{and} \qquad \mathsf{ct} := [1]^L.$$

Finally, when $\mathcal{A}$ outputs its guess b' to $\mathcal{B}$, then $\mathcal{B}$ also outputs b' as solution to the q-DBDHI instance.

Analysis of $\mathcal{B}$. Recapping the construction of $\mathcal{B}$, we observe that mpk and all answers of $\mathcal{B}$ are distributed identically to the challenger's interactions with $\mathcal{A}$ in Game 4. Analyzing $\mathcal{B}$'s response to the challenge id^*, we distinguish the following two cases depending on the q-DBDHI instance.
Let $V = e([y],[y])^{1/\alpha}$. Then we have for the first part of (14)

$$\begin{aligned} V^{\zeta\cdot\varphi_0} e\left([y], \prod_{k=1}^{q-4\lambda-8} [y\alpha^{k-1}]^{\zeta\varphi_k}\right) &= e([y],[y])^{\zeta\cdot\varphi_0/\alpha} e\left([y], \prod_{k=1}^{q-4\lambda-7} [y\alpha^{k-1}]^{\zeta\varphi_k}\right) \\ = e\left([y], [y \sum_{k=0}^{q-4\lambda-8} \zeta\varphi_k \alpha^{k-1}]\right) &= e\left([y],[y]\right)^{\zeta\mathsf{Q}(\alpha)/\alpha} \end{aligned}$$

and for the second part

$$e\left(\prod_{k=0}^{q} [y\alpha^k]^{\gamma_k}, \prod_{k=0}^{q-4\lambda-8} [y\alpha^k]^{\varphi_k}\right) = e\left([y\mathsf{R}(\alpha)], [y\mathsf{Q}(\alpha)]\right) = e\left([y],[y]\right)^{\mathsf{R}(\alpha)\mathsf{Q}(\alpha)}.$$

Together we have that

$$\tilde{K} = e\left([y],[y]\right)^{\zeta\cdot\mathsf{Q}(\alpha)/\alpha + \mathsf{R}(\alpha)\cdot\mathsf{Q}(\alpha)} = e\left([y],[y]\right)^{\mathsf{Q}(\alpha)^2/\mathsf{P}_{\mathsf{id}^*}(\alpha)} = e\left([1],[1]\right)^{1/P_{\mathsf{id}^*}(\alpha)},$$

where we use $[y\mathsf{Q}(\alpha)] = [1]$ and Lemma 12 multiplied by $\mathsf{Q}(\mathsf{Z})$ on both sides.
Then if we implicitly set $s := L/\mathsf{P}_{\mathsf{id}^*}(\alpha)$ we have

$$K = \tilde{K}^L = e\left([1],[1]\right)^{L/\mathsf{P}_{\mathsf{id}^*}(\alpha)} = e\left([1],[1]\right)^s \text{ and } \mathsf{ct} = [1]^L = [1]^{\mathsf{P}_{\mathsf{id}^*}(\alpha)\cdot s}.$$

Since $\mathsf{P}_{\mathsf{id}^*}(\alpha) = u(\mathsf{id}^*)$ we have that ct is a correct encapsulation of K.

In case that $V \xleftarrow{\$} \mathbb{G}_T$ we have that $\tilde{K}^L$ is uniformly random in $\mathbb{G}_T$. Thus $K = \tilde{K}$ and ct do not match. Overall it can be seen that $\mathcal{B}$ simulates Game Game 4 perfectly. Plugging the game sequence and the reduction together gives us the same result as in the proof of Theorem 4. That is

$$\mathsf{Adv}_{\mathcal{B}}^{q\text{-DBDHI}}(\lambda) \geq \varepsilon_{\mathcal{A}}^2/(32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}) - \mathsf{negl}(\lambda).$$

Running time of $\mathcal{B}$. The running time $t_{\mathcal{B}}$ of $\mathcal{B}$ consists of the running time $t_{\mathcal{A}}$ of $\mathcal{A}$ plus the time required to compute a valid public key mpk and the time to respond to oracle queries by $\mathcal{A}$. For computing mpk and these responses the most time consuming operations are exponentiation. Each computation needs at most q exponentiations. By Lemma 11 we have $q \leq 4\lambda + 8 + \lfloor \log(2\lambda + 3) \rfloor + 32t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}} = O(t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}})$ and thus

$$t_{\mathcal{B}} = t_{\mathcal{A}} + O(q) = O(t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}}).$$

This completes the proof. □

5 Verifiable Random Functions from Pairings

In this section, we use blockwise partitioning in order to construct the first verifiable random function without random oracles that has both, short proofs and short public keys. Compared to previous VRF constructions that also achieve small proof sizes, like [Kat17, Koh19], we achieve much better concrete proof sizes or much smaller public keys. Concretely, we achieve public keys consisting of only 10 group elements and proofs consisting of only 9 group elements for $\lambda = 128$. We start by introducing preliminaries on VRFs and then present our construction.

5.1 Preliminaries on VRFs

Syntax of VRFs. Formally, a VRF consists of algorithms $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ with the following syntax.

- $(\mathsf{vk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ takes as input a security parameter λ and outputs a key pair $(\mathsf{vk}, \mathsf{sk})$. We say that sk is the *secret key* and vk is the *verification key*.
- $(Y, \pi) \xleftarrow{\$} \mathsf{Eval}(\mathsf{sk}, X)$ takes as input a secret key sk and $X \in \{0,1\}^\lambda$, and outputs a function value $Y \in \mathcal{Y}$, where $\mathcal{Y}$ is a finite set, and a proof π. We write $V_{\mathsf{sk}}(X)$ to denote the function value Y computed by Eval on input (sk, X).
- $\mathsf{Vfy}(\mathsf{vk}, X, Y, \pi) \in \{0,1\}$ takes as input a verification key vk, $X \in \{0,1\}^\lambda$, $Y \in \mathcal{Y}$, and proof π, and outputs a bit.

$\underline{\mathsf{RoR}^{\Theta}_{\mathcal{A}}(\lambda)}$
$b \xleftarrow{\$} \{0,1\}$
$(\mathsf{vk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$
$(X^*, st) \leftarrow \mathcal{A}_1^{\mathsf{Eval}(\mathsf{sk},\cdot)}(1^\lambda, \mathsf{vk})$
$(Y_0, \pi) \xleftarrow{\$} \mathsf{Eval}(\mathsf{sk}, X^*)$; $Y_1 \xleftarrow{\$} \mathcal{Y}$
$b' \leftarrow \mathcal{A}_2^{\mathsf{Eval}(\mathsf{sk},\cdot)}(Y_b)$
If $(b' == b)$ return 1, else 0

Figure 3: The real-or-random(RoR) experiment for VRFs, executed with scheme $\mathcal{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ and adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracle $\mathsf{Eval}(\mathsf{sk}, X)$ returns (Y, π) with the restriction that $\mathcal{A}$ is not allowed to query oracle $\mathsf{Eval}(\mathsf{sk}, \cdot)$ for the challenge query X^*.

Definition 10. $\mathcal{VRF} := (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ is a *verifiable random function* (VRF) if all of the following hold.

Correctness. For all $(\mathsf{vk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ and $X \in \{0,1\}^\lambda$ holds: if $(Y, \pi) \xleftarrow{\$} \mathsf{Eval}(\mathsf{sk}, X)$, then we have $\mathsf{Vfy}(\mathsf{vk}, X, Y, \pi) = 1$. Algorithms Gen, Eval, Vfy are polynomial-time.

$$\Pr\left[\begin{array}{ll} (\mathsf{vk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda) & \\ X \in \{0,1\}^\lambda & : \mathsf{Vfy}(\mathsf{vk}, X, Y, \pi) = 1 \\ (Y, \pi) \xleftarrow{\$} \mathsf{Eval}(\mathsf{sk}, X) & \end{array}\right] = 1$$

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

Pseudorandomness. Consider an attacker $\mathcal{A}$ with access (via oracle queries) to the procedures defined in Figure 3. We say that $\mathcal{A}$ is *legitimate*, if $\mathcal{A}$ never queries $\mathsf{Evaluate}(X^*)$, where X^* is the output of $\mathcal{A}_1$. We define the advantage of $\mathcal{A}$ in breaking the pseudorandomness of VRF $\mathcal{VRF}$ as

$$\mathsf{Adv}^{\mathsf{RoR}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{RoR}^{\mathcal{VRF}}_{\mathcal{A}}(\lambda) = 1\right] - 1/2\right|$$

In order to fulfill the unique provability requirement of VRFs, we need that each group element has a unique encoding. Otherwise two different encodings of the same group element could be accepted by the Vfy algorithms and by that breaking the unique provability requirement. We therefore require the group generator to be certified as defined by Hofheinz and Jager [HJ16].

Definition 11. We say that group generator GrpGen is *certified*, if there exist deterministic polynomial-time (in the security parameter) algorithms GrpVfy and GrpElemVfy with the following properties.

Parameter Validation. Given the security parameter (in unary) and a string $\mathcal{BG}$, which is not necessarily generated by GrpGen, algorithm $\mathsf{GrpVfy}(1^\lambda, \mathcal{BG})$ outputs 1 *if and only if* $\mathcal{BG}$ has the form

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

and all requirements from Definition 7 are satisfied.

Recognition and Unique Representation of Elements of $\mathbb{G}$. We also require that each element in $\mathbb{G}$ has a *unique* representation, which can be efficiently recognized. That is, on input the security parameter (in unary) and two strings $\mathcal{BG}$ and s, $\mathsf{GrpElemVfy}(1^\lambda, \mathcal{BG}, s)$ outputs 1 if and only if $\mathsf{GrpVfy}(1^\lambda, \mathcal{BG}) = 1$ 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 $\mathcal{BG}$ to specify the representation of elements of $\mathbb{G}$.

5.2 A VRF with Short Proofs and Keys

The construction of the VRF roughly follows the construction of the IB-KEM. The major difference is that including group elements in the proof of the VRF allows us to only include the group elements $[x_i]$ instead of all possible combinations $F(\mathsf{msk}, m)$, as in the IB-KEM, in the public keys. Hence, the proofs and secret keys of VRF construction consist of only $\lfloor \log(2\lambda + 3) \rfloor + 1$ many elements and the secret keys consist of only $\lfloor \log(2\lambda + 3) \rfloor + 2$ group elements. Instead of viewing the VRF as a adaptation of our previous IB-KEM in Section 4.2, it can also be viewed as an adaptation of Yamada's VRF [Yam17a] to blockwise partitioning and the description mostly follows Yamada's VRF. Let $\mathcal{H}_\lambda = \{H : \{0,1\}^* \to \{0,1\}^{2\lambda+3}\}$ be a family of hash functions, let $\ell = \lfloor \log(2\lambda + 3) \rfloor$, let GrpGen be a certified bilinear group generator, and let $\mathcal{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ be the following algorithms.

Key generation. $\mathsf{Gen}(1^\lambda)$ chooses a group description $\mathcal{BG} \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$, a random hash function $H \xleftarrow{\$} \mathcal{H}_\lambda$, a random generator $[1] \xleftarrow{\$} \mathbb{G}^*$. Then it samples $w_i \xleftarrow{\$} \mathbb{Z}_p^*$ and sets $W_i := [w_i]$ for all $i = 0, \ldots, \ell$. It returns

$$\mathsf{vk} := ([1], \mathcal{BG}, W_0, \ldots, W_\ell, H) \qquad \text{and} \qquad \mathsf{sk} := (w_0, \ldots, w_\ell).$$

Evaluation. $\mathsf{Eval}(\mathsf{sk}, X)$ computes for $i = 0, \ldots, \ell$

$$\Theta_i(X) := \prod_{i'=0}^{i} (w_{i'} + H_{i'}(X)).$$

If there is an index $0 \leq i \leq \ell$ such that $\Theta_i(X) \equiv 0 \bmod p$ it sets $Y := 1_{\mathbb{G}_T}$ and $\pi_i = 1_{\mathbb{G}}$ for all $i = 0, \ldots, \ell$. Otherwise, it computes

$$Y := e([1], [1])^{1/\Theta_\ell(X)} \qquad \text{and} \qquad \pi_i := g^{1/\Theta_i(X)}$$

for all $i = 0, \ldots, \ell$. It outputs $(Y, \pi = (\pi_0, \ldots, \pi_\ell))$.

Verification. $\mathsf{Vfy}(\mathsf{vk}, X, Y, \pi)$ checks if the following conditions are met and outputs 0 if not, otherwise it outputs 1.

1. We have that $X \in \{0,1\}^*$.
2. vk has the form $([1], \mathcal{BG}, W_0, \ldots, W_\ell, H)$ and $\mathsf{sk} = (w_0, \ldots, w_\ell)$.
3. $\mathcal{BG}$ is a a certified encoding of a bilinear group: $\mathsf{GrpVfy}(1^\lambda, \mathcal{BG}) = 1$. Further, all group elements can be verified by running $\mathsf{GrpElemVfy}(1^\lambda, \mathcal{BG}, [1]) = 1$, $\mathsf{GrpElemVfy}(1^\lambda, \mathcal{BG}, h) = 1$, $\mathsf{GrpElemVfy}(1^\lambda, \mathcal{BG}, W_i) = 1$ and also $\mathsf{GrpElemVfy}(1^\lambda, \mathcal{BG}, \pi_i) = 1$ for all $0 \leq i \leq \ell$.
4. If there is an index $\leq i \leq \ell$ such that $W_i \cdot [H_i(X)] = 1_{\mathbb{G}}$, then it holds that $Y = 1_{\mathbb{G}_T}$ and $\pi_i = 1_{\mathbb{G}}$ for all $i = 0, \ldots, \ell$.
5. If we have $W_i \cdot [H_i(X)] \neq 1_{\mathbb{G}}$ for all $i = 0, \ldots, \ell$, then for all of these i it holds that $e(\pi_i, W_i \cdot [H_i(X)]) = e([1], \pi_{i-1})$.
6. It holds that $e(\pi_\ell, [1]) = Y$.

$\mathcal{VRF}$ as specified above is correct and fulfills the unique provability requirements as can be proven with standard arguments. Also note that using a hash function does not affect unique provability because the hash function deterministically maps each input to an output. Like the IB-KEM we present in Section 4.2, our VRF is based on the q-DBDHI assumption. We set $q := \log(2\lambda + 3) + 2 + 2 \sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} \left(2^{2^i} - 1\right)$ which is at most $\log(2\lambda + 3) + 2 + \frac{32t - \mathcal{A}^2}{\varepsilon_\mathcal{A}}$ by Lemma 11.

Theorem 4. *If $\mathcal{VRF}$ is instantiated with a family $\mathcal{H} = \{H : \{0,1\}^* \to \{0,1\}^{2\lambda+3}\}$ of weak near-collision resistant hash functions from Definition 1, then for any legitimate attacker $\mathcal{A}$ that breaks the pseudorandomness of $\mathcal{VRF}$ in time $t_\mathcal{A}$ with advantage $\varepsilon_\mathcal{A} := \mathsf{Adv}^{\mathsf{RoR}}_\mathcal{A}(\lambda)$, there exists an algorithm $\mathcal{B}$ that, given (sufficiently close approximations of) $t_\mathcal{A}$ and $\varepsilon_\mathcal{A}$, breaks the q-DBDHI assumption with $q \leq \lfloor \log(2\lambda + 3) \rfloor + 2 + 32t_\mathcal{A}^2/\varepsilon_\mathcal{A}$ in time $t_\mathcal{B} = O(t_\mathcal{A}^2/\varepsilon_\mathcal{A})$ and with*

$$\mathsf{Adv}^{q\text{-DBDHI}}_\mathcal{A}(\lambda) \geq \varepsilon_\mathcal{A}^2/(32t_\mathcal{A}^2 - 16t_\mathcal{A}) - \mathsf{negl}(\lambda),$$

for some negligible term negl.

The proof of Theorem 4 follows basically the proof of Theorem 3. For a complete overview, we provide it in Appendix D.

References

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, Heidelberg, May / June 2010. 4, 7, 3.2, 3.3, 3.3, 4, 3.3, 8, 3.3

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 98–115. Springer, Heidelberg, August 2010. 1.1, 4

[AFL16] Daniel Apon, Xiong Fan, and Feng-Hao Liu. Compact identity based encryption from LWE. Cryptology ePrint Archive, Report 2016/125, 2016. `http://eprint.iacr.org/2016/125`. 1.1

[AHY15] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. A framework for identity-based encryption with almost tight security. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 521–549. Springer, Heidelberg, November / December 2015. 1.1

[Alp15] Jacob Alperin-Sheriff. Short signatures with short public keys from homomorphic trapdoor functions. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 236–255. Springer, Heidelberg, March / April 2015. 1.1, 2.1

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, Heidelberg, May 2004. 1, 1.1, 2, 4, 4.1, 8, 4.2, 9

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459. Springer, Heidelberg, August 2004. 1

[BC04] Eli Biham and Rafi Chen. Near-collisions of SHA-0. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 290–305. Springer, Heidelberg, August 2004. 1.1

[BCJ+05] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and reduced SHA-1. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 36–57. Springer, Heidelberg, May 2005. 1.1

[BDPV08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 181–197. Springer, Heidelberg, April 2008. 1.1

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001. 1, 1.1, 2

[BFMS08] Kamel Bentahar, Pooya Farshim, John Malone-Lee, and Nigel P. Smart. Generic constructions of identity-based and certificateless KEMs. *Journal of Cryptology*, 21(2):178–199, April 2008. 3.4

[BHJ+13] Florian Böhl, Dennis Hofheinz, Tibor Jager, Jessica Koch, Jae Hong Seo, and Christoph Striecks. Practical signatures from standard assumptions. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 461–485. Springer, Heidelberg, May 2013. 2.1, 2.1, 2.1.1, 2.1.1

[BHJ+15] Florian Böhl, Dennis Hofheinz, Tibor Jager, Jessica Koch, and Christoph Striecks. Confined guessing: New signatures from standard assumptions. *Journal of Cryptology*, 28(1):176–208, January 2015. 1.1, 2.1, 2.1, 2.1.1, 2.1.1

[BL16] Xavier Boyen and Qinyi Li. Towards tightly secure lattice short signature and id-based encryption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 404–434. Springer, Heidelberg, December 2016. 1.1

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013. 3.1, 4

[BMR10] Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010*, pages 131–140. ACM Press, October 2010. 1.1

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 499–517. Springer, Heidelberg, May 2010. 1.1, 3.2

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993. 1

[BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424. Springer, Heidelberg, April 2009. 1, 1.1, 1.1, 4

[BV16] Zvika Brakerski and Vinod Vaikuntanathan. Circuit-ABE from LWE: Unbounded attributes and semi-adaptive security. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 363–384. Springer, Heidelberg, August 2016. 3.1

[CFN15] Dario Catalano, Dario Fiore, and Luca Nizzardo. Programmable hash functions go private: Constructions and applications to (homomorphic) signatures with shorter public keys. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 254–274. Springer, Heidelberg, August 2015. 1

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998. 1

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, Heidelberg, May / June 2010. 1.1, 4

[CLL+13] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter ibe and signatures via asymmetric pairings. In Michel Abdalla and Tanja Lange, editors, *Pairing-Based Cryptography – Pairing 2012*, pages 122–140, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 1.1

[DM14] Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 335–352. Springer, Heidelberg, August 2014. 1.1, 2.1

[FF13] Marc Fischlin and Nils Fleischhacker. Limitations of the meta-reduction technique: The case of Schnorr signatures. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 444–460. Springer, Heidelberg, May 2013. 1

[FHJ20] Marc Fischlin, Patrick Harasser, and Christian Janson. Signatures from sequential-OR proofs. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 212–244. Springer, Heidelberg, May 2020. 2.1.1

[FHPS13] Eduarda S. V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 513–530. Springer, Heidelberg, August 2013. 1

[FLR+10] Marc Fischlin, Anja Lehmann, Thomas Ristenpart, Thomas Shrimpton, Martijn Stam, and Stefano Tessaro. Random oracles with(out) programmability. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 303–320. Springer, Heidelberg, December 2010. 1, 2.1.1

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 445–464. Springer, Heidelberg, May / June 2006. 1.1

[Gol08] Oded Goldreich. *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008. 1, 3

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 4

[HJ16] Dennis Hofheinz and Tibor Jager. Verifiable random functions from standard assumptions. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 336–362. Springer, Heidelberg, January 2016. 1.1, 7, 5.1

[HJK11] Dennis Hofheinz, Tibor Jager, and Eike Kiltz. Short signatures from weaker assumptions. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 647–666. Springer, Heidelberg, December 2011. 1

[HK08] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 21–38. Springer, Heidelberg, August 2008. 1

[HMS12] Goichiro Hanaoka, Takahiro Matsuda, and Jacob C. N. Schuldt. On the impossibility of constructing efficient key encapsulation and programmable hash functions in prime order groups. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 812–831. Springer, Heidelberg, August 2012. 1

[HW10] Susan Hohenberger and Brent Waters. Constructing verifiable random functions with large input spaces. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 656–672. Springer, Heidelberg, May / June 2010. 1.1

[Jag15] Tibor Jager. Verifiable random functions from weaker assumptions. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 121–143. Springer, Heidelberg, March 2015. 1.1, 1.1

[JK18] Tibor Jager and Rafael Kurek. Short digital signatures and ID-KEMs via truncation collision resistance. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 221–250. Springer, Heidelberg, December 2018. 1, 1.1, 1.1, 1.1, 2.1, 2.1, 2.1.1, 2.1.1, 2.1.1, 3.5, 4, 4.2, A, D

[JN19] Tibor Jager and David Niehues. On the real-world instantiability of admissible hash functions and efficient verifiable random functions. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019*, volume 11959 of *LNCS*, pages 303–332. Springer, Heidelberg, August 2019. 1, 1.1, 1.1, 3, 2.1.1, 3.5, A, D

[Kat17] Shuichi Katsumata. On the untapped potential of encoding predicates by arithmetic circuits and their applications. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 95–125. Springer, Heidelberg, December 2017. 1.1, 1.1, 5

[Koh19] Lisa Kohl. Hunting and gathering - verifiable random functions from standard assumptions with short proofs. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 408–437. Springer, Heidelberg, April 2019. 1.1, 1.1, 5

[KY16] Shuichi Katsumata and Shota Yamada. Partitioning via non-linear polynomial functions: More compact IBEs from ideal lattices and bilinear maps. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 682–712. Springer, Heidelberg, December 2016. 1.1, 3

[Lew12] Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 318–335. Springer, Heidelberg, April 2012. 1.1

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012. 3.1, 5, 6, 3.2

[MvOV96] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 5th edition, 1996. 1.1

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126. Springer, Heidelberg, August 2002. 1

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009. 3.1

[Pei15] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. http://eprint.iacr.org/2015/939. 1.1

[PS14] Inna Polak and Adi Shamir. Using random error correcting codes in near-collision attacks on generic hash-functions. In Willi Meier and Debdeep Mukhopadhyay, editors, *INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 219–236. Springer, Heidelberg, December 2014. 1.1

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. 2, 3.1, 2, 3.1, 7

[Ros18] Razvan Rosie. Adaptive-secure VRFs with shorter keys from static assumptions. In Jan Camenisch and Panos Papadimitratos, editors, *CANS 18*, volume 11124 of *LNCS*, pages 440–459. Springer, Heidelberg, September / October 2018. 1.1

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/2004/332. 3.5, D

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, Heidelberg, May 2005. 1, 1.1, 1.1, 4, 4.1, 4.1

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, August 2009. 1.1

[Yam16] Shota Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 32–62. Springer, Heidelberg, May 2016. 1.1

[Yam17a] Shota Yamada. Asymptotically compact adaptively secure lattice IBEs and verifiable random functions via generalized partitioning techniques. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 161–193. Springer, Heidelberg, August 2017. 1.1, 1, 1.1, 1.1, 1.1, 3, 3.2, 3.4, 3.4, 4, 12, 4, 5.2, D

[Yam17b] Shota Yamada. Asymptotically compact adaptively secure lattice IBEs and verifiable random functions via generalized partitioning techniques. Cryptology ePrint Archive, Report 2017/096, 2017. http://eprint.iacr.org/2017/096. 1.1, 3.4, 1, 3.5, 2

[ZCZ16] Jiang Zhang, Yu Chen, and Zhenfeng Zhang. Programmable hash functions from lattices: Short signatures and IBEs with small key sizes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 303–332. Springer, Heidelberg, August 2016. 1.1, 1.1

[Zha16] Mark Zhandry. The magic of ELFs. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 479–508. Springer, Heidelberg, August 2016. 1, 2.1.1

A Proof of Identities and Inequalities from [JK18, JN19]

We start by proving $n' \in \{1, \ldots, 2\lambda + 3\}$.

$$\begin{aligned} n' = \lceil \log(4t(2t-1)/\varepsilon) \rceil &\leq \left\lceil \log\left(4 \cdot 2^{\lambda}(2t-1)\right) \right\rceil \\ &\leq \left\lceil \log\left(8 \cdot 2^{\lambda} t\right) \right\rceil \leq \left\lceil \log\left(2^{\lambda} 2^{\lambda+3}\right) \right\rceil = 2\lambda + 3 \end{aligned}$$

Since $4t(2t-1) = 8t^2 - 4t > 1$ for all $t \in \mathbb{N}$ and $\varepsilon \in (0, 1]$, we have $\log(4t(2t-1)/\varepsilon) > 0$ and therefore $j \geq 1$.

We proceed to prove $2t(2t-1)/2^{n'} \leq \varepsilon/2$.

$$\frac{2t(2t-1)}{2^{n'}} = \frac{2t(2t-1)}{2^{\lceil \log(4t(2t-1)/\varepsilon) \rceil}} \leq \frac{\varepsilon 2t(2t-1)}{4t(2t-1)} = \frac{\varepsilon}{2}$$

Finally, we have

$$\frac{1}{2^{n'}} = \frac{1}{2^{\lceil \log(4t(2t-1)/\varepsilon) \rceil}} \geq \frac{1}{2} \cdot \frac{\varepsilon}{4t(2t-1)} = \frac{\varepsilon}{16t^2 - 8t}.$$

B Proof of Lemma 11

$$\begin{aligned} 2 \cdot \sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} \left(2^{(2^i)} - 1\right) < 2 \cdot \sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} 2^{(2^i)} &< 2 \cdot \prod_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} 2^{(2^i)} \qquad (15) \\ = 2 \cdot 2^{\sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \perp}} (2^i)} &= 2^{\ell} \\ = 2 \cdot 2^{\lceil \log(4t_{\mathcal{A}}(2t_{\mathcal{A}}-1)/\varepsilon_{\mathcal{A}}) \rceil} &\leq \frac{8t_{\mathcal{A}}(2t_{\mathcal{A}}-1)}{\varepsilon_{\mathcal{A}}} \\ \geq 2 \cdot \frac{16t_{\mathcal{A}}^2}{\varepsilon_{\mathcal{A}}} &= \frac{32t_{\mathcal{A}}^2}{\varepsilon_{\mathcal{A}}}, \end{aligned}$$

where Inequality (15) holds, because $a + b < ab$ for all $a, b \geq 2$. This concludes the proof.

C Proof of Lemma 12

In order to decomposition $\mathsf{P}_X(\mathsf{Z})$ and $\mathsf{Q}(\mathsf{Z})$, we define

$$t_X := \left(\prod_{\substack{i \in \{0, \ldots, \lfloor \log(2\lambda+3) \rfloor\} \\ K_i \neq \perp}}^{n} (\tilde{w}_i + H_i(X)) \right)$$

and polynomials $\mathsf{S}_{X,i}(\mathsf{Z}), \mathsf{Q}_i(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ for all $i \in \{0, \ldots, \lfloor \log(2\lambda+3) \rfloor\}$ as

$$\mathsf{S}_{X,i}(\mathsf{Z}) := \tilde{w}_i \mathsf{Z} - K_i + H_i(X) \qquad \text{and} \qquad \mathsf{Q}_i(\mathsf{Z}) = \prod_{\substack{-2^{2^i}+1 \le k \le 2^{2^i}-1 \\ k \neq 0}} (\tilde{w}_i \mathsf{Z} + k)\,,$$

where we only define $\mathsf{S}_{X,i}(\mathsf{Z})$ for i such that $K_i \neq \bot$. Recall that we denote the number of non-wildcard positions in K by $j \in \{1, \ldots, \lfloor \log(2\lambda+3) \rfloor + 1\}$. By the definitions of $\mathsf{P}_X(\mathsf{Z})$ and $\mathsf{Q}(\mathsf{Z})$, we then have that

$$\mathsf{P}_X(\mathsf{Z}) = t_X \prod_{\substack{i \in \{0,\ldots,\lfloor \log(2\lambda+3) \rfloor\} \\ i \neq \bot}}^{j} \mathsf{S}_{X,i}(\mathsf{Z}) \qquad \text{and} \qquad \mathsf{Q}(\mathsf{Z}) = \mathsf{Z}^{j-1} \prod_{i=0}^{\lfloor \log(2\lambda+3) \rfloor} \mathsf{Q}_i(\mathsf{Z})$$

holds. Now let $I := \{i \in \{0, \ldots, j\} : K_i \neq \bot \wedge H_i(X) = K_i\}$. We first observe, that for all $i \in I$ it holds that $\mathsf{S}_{X,i}(\mathsf{Z}) = \tilde{w}_i \mathsf{Z} - K_i + H_i(X) = \tilde{w}_i \mathsf{Z}$. Therefore, it follows from the definition of $\mathsf{Q}(\mathsf{Z})$ that $\mathsf{S}_{X,i}(\mathsf{Z}) \nmid \mathsf{Q}_i(\mathsf{Z})$ for all $i \in I$.

We first observe that $K_i = H_i(X)$ *if and only if* $|I| = j$. Using the previous observation, it follows that $P_X(\mathsf{Z}) \nmid \mathsf{Q}(\mathsf{Z})$ but $P_X(\mathsf{Z}) \mid \mathsf{Z} \cdot \mathsf{Q}(\mathsf{Z})$ because $\mathsf{P}_X(\mathsf{Z}) = t_X \mathsf{Z}^j \prod_{i=1}^{j} \tilde{w}_i$ by definition. This implies the existence of a polynomial $\mathsf{R}_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ and $\zeta_X \in \mathbb{Z}_p$ such that $\mathsf{Q}(\mathsf{Z}) = \zeta_X/\mathsf{Z} + \mathsf{R}_X(\mathsf{Z})$.

We now consider the case that there exists $i \in \{0, \ldots, \lfloor \log(2\lambda+3) \rfloor\}$ such that $K_i \neq \bot$ and $H(X)_i \neq K_{i'}$. Let $L := -K_i + H_i(X)$ and note that $L \neq 0$ and $-2^{2^i} + 1 \le L \le 2^{2^i} - 1$. Furthermore, it holds that $\mathsf{S}_{X,i}(\mathsf{Z}) = \tilde{w}_i \mathsf{Z} - K_i + H_i(X) = \tilde{w}_i \mathsf{Z} + L$ by the definition of $\mathsf{S}_{X,i}(\mathsf{Z})$. In particular, it therefore holds that $\mathsf{S}_{X,i'}(\mathsf{Z}) \mid \mathsf{Q}_{i'}(\mathsf{Z})$. We therefore also have that $\mathsf{P}_X(\mathsf{Z}) \mid \mathsf{Q}(\mathsf{Z})$ and hence it exists a polynomial $R_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $R_X(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}] = \mathsf{Q}(\mathsf{Z})/\mathsf{P}_X(\mathsf{Z})$. This completes the proof Lemma 12.

D Proof of Theorem 4

PROOF. We prove Theorem 4 with a sequence of games argument [Sho04]. The first half of the proof follows the proofs by Jager, Kurek and Niehues [JK18, JN19]. The second half follows the proof by Yamada [Yam17a]. We denote with G_i the event that Game i outputs 1 and with $E_i := \Pr\left[1 \xleftarrow{\$} G_i\right] - 1/2$ the advantage of $\mathcal{A}$ in Game i.

Game 0. This is the original VRF security experiment. We therefore have

$$E_0 = \Pr\left[\mathsf{RoR}_{\mathcal{A}}^{\mathcal{VRF}}(\lambda) = 1\right] - 1/2 = \varepsilon_{\mathcal{A}}.$$

Game 1. This game is identical to Game 0, except that the challenger runs $\bar{K} = (K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(t_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$ from Lemma 1. Furthermore, it defines $\mathcal{I} := \{i : K_i \neq \bot\}$. Let $\mathcal{Q}$ be the set of all queries that the adversary makes to $\mathsf{Eval}(\mathsf{sk}, X^{(i)})$, and let $\mathcal{Q}^* := \mathcal{Q} \cup \{X^*\}$, where X^* is the challenge query. Additionally, the challenger raises event coll, aborts and outputs a random bit if there exist $X, X' \in \mathcal{Q}$ such that $X \neq X'$, but $H_i(X) = H_i(X')$ for all $i \in \mathcal{I}$. Since coll is defined exactly as in Lemma 1 we have

$$E_1 \ge E_0 - \Pr[\mathsf{coll}] = \varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}]\,.$$

Game 2. In this game, the challenger raises event $\mathsf{badChal}$ which occurs if there exists an index $i \in \mathcal{I}$ such that $H_i(X^*) \neq K_i$ and it raises event $\mathsf{badEval}$ if there exists $X \in \mathcal{Q}$ such that $H_i(X) = K_i$ for all $i \in \mathcal{I}$. If $\mathsf{badChal}$ or $\mathsf{badEval}$ occur it aborts and outputs a random bit. By Property 2 of Lemma 1 we have $\mathsf{badEval} \implies \mathsf{coll} \vee \mathsf{badChal}$ and by Property 1 we have

$$E_2 = E_1 \cdot \Pr[\neg\mathsf{badChal}] = (\varepsilon_{\mathcal{A}} - \Pr[\mathsf{coll}]) \cdot \Pr[\neg\mathsf{badChal}] \geq \frac{\varepsilon_{\mathcal{A}}^2}{32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}}$$

Game 3. In this game the challenger changes the way it generates vk . It samples $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$, $\tilde{w}_i \xleftarrow{\$} \mathbb{Z}_p^*$ and sets for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$

$$w_i := \begin{cases} \tilde{w}_i \cdot \alpha - K_i & \text{if } K_i \neq \bot \\ \tilde{w}_i & \text{otherwise.} \end{cases}$$

If $w_i = 0$ for any $0 \leq i \leq \lfloor \log(2\lambda + 3) \rfloor$, then the challenger aborts and outputs a random bit. This happens *iff* $\tilde{w}_i \alpha = K_i$. Since $\tilde{w}_i \alpha$ is distributed uniformly at random in $\mathbb{Z}_p^*$, the probability that this happens for any of the $\mathcal{O}(\log \lambda)$ many w_i is negligible and we therefore have $E_3 = E_2 - \mathsf{negl}(\lambda)$.

Game 4. In this game, we change the way the challenger chooses $[1]$. Let $j = \mathcal{I}|$ be the number of non-wildcard positions in $\bar{K}$. The challenger then first defines the polynomial $\mathsf{Q}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ as

$$\mathsf{Q}(\mathsf{Z}) := \mathsf{Z}^{j-1} \prod_{\substack{i=0 \\ K_i \neq \bot}}^{\lfloor \log(2\lambda+3) \rfloor} \prod_{\substack{-2^{2^i}+1 \leq k \leq 2^{2^i}-1 \\ k \neq 0}} (\tilde{w}_i \mathsf{Z} + k)\,. \tag{16}$$

The challenger samples $[y] \xleftarrow{\$} \mathbb{G}_1$ and sets $[1] := [y]^{\mathsf{Q}(\alpha)}$. If $[1] = 1_{\mathbb{G}}$, which happens *iff* $\mathsf{Q}(\alpha) \equiv 0 \bmod p$, the challenger outputs a random bit and aborts. It can be seen that the distribution of $[1]$ changes only if $\mathsf{Q}(\alpha) \equiv 0 \bmod p$. By Lemma 11 we have $\deg(\mathsf{Q}) \leq \lfloor \log(2\lambda + 3) \rfloor + 2 + 32t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}}$. Since α is uniformly random in $\mathbb{Z}_p^*$, we have by the Schwartz-Zippel Lemma $\Pr[\mathsf{Q}(\alpha) = 0] \leq \frac{\deg(\mathsf{Q})}{p-1}$. Due to the fact that $\deg(\mathsf{Q})$ is polynomial in λ and $p \in 2^{\Omega(\lambda)}$ by the properties of GrpGen, we have $E_4 = E_3 - \mathsf{negl}(\lambda)$.
Now we are ready to construct $\mathcal{B}$, which simulates Game 4 as follows.

Initialization. At the beginning of the experiment $\mathcal{B}$ runs $\bar{K} = (K_0, \ldots, K_\ell) \xleftarrow{\$} \mathsf{BPSmp}(t_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$ and by doing so it also determines the index set $\mathcal{I} = \{i : K_i \neq \bot\}$ according to Lemma 1. Moreover, it receives a q-DBDHI instance $\left(\mathcal{BG}, [y], [y\alpha], [y\alpha^2], \ldots, [y\alpha^q], V\right)$, where $q = j + \sum_{\substack{i \in [\lfloor \log(2\lambda+3) \rfloor]_0 \\ K_i \neq \bot}} \left(2^{2^i} - 1\right)$ and $j = |\mathcal{I}| \in \{0, \ldots, \ell\}$ is the number of non-wildcard positions in $\bar{K}$. Further we either have $V = e([y], [y]s)^{1/\alpha}$ or $V \xleftarrow{\$} \mathbb{G}_T$. In order to define $[1]$, $\mathcal{B}$ samples $\tilde{w}_i \xleftarrow{\$} \mathbb{Z}_p^*$ for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$. Then it computes coefficients $\varphi_0, \ldots \varphi_{q-1} \in \mathbb{Z}_p$

$$\mathsf{Q}(\mathsf{Z}) = \mathsf{Z}^{j-1} \prod_{\substack{i=0 \\ K_i \neq \bot}}^{\lfloor \log(2\lambda+3) \rfloor} \prod_{\substack{-2^{2^i}+1 \leq k \leq 2^{2^i}-1 \\ k \neq 0}} (\tilde{w}_i \mathsf{Z} + k) = \sum_{k=0}^{q-1} \varphi_k \mathsf{Z}^k,$$

where $\mathsf{Q}(\mathsf{Z})$ is as in Equation (16). Then $\mathcal{B}$ defines

$$[1] := \prod_{k=0}^{q-1} [y\alpha^k]^{\varphi_k} = [y \sum_{k=0}^{q-1} \varphi_k \alpha^k] = [y\mathsf{Q}(\alpha)].$$

If $[1] = 1_{\mathbb{G}}$, meaning $\mathsf{Q}(\alpha) \equiv 0 \bmod p$, $\mathcal{B}$ aborts and outputs a random bit. $\mathcal{B}$ proceeds with computing vk. First $\mathcal{B}$ samples $\tilde{w}_i$ for $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$ and then it (implicitly) sets

$$w_i := \begin{cases} \tilde{w}_i \cdot \alpha - K_i & \text{if } K_i \neq \perp \\ \tilde{w}_i & \text{otherwise.} \end{cases}$$

by computing $\psi_{i,0}, \ldots, \psi_{i,q} \in \mathbb{Z}_p$ for all $i = 0, ..., \lfloor \log(2\lambda + 3) \rfloor$ such that

$$\mathsf{Q}(\alpha) \cdot w_i = \sum_{k=0}^{q} \psi_{i,k} \alpha^k.$$

and setting

$$W_i := \prod_{k=0}^{q} [y\alpha^k]^{\psi_{i,k}} = [y\mathsf{Q}(\alpha) \cdot w_i] = [w_i].$$

for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$. Finally, $\mathcal{B}$ gives $\mathsf{vk} := ([1], \mathcal{BG}, W_0, \ldots, W_{\lfloor \log(2\lambda+3) \rfloor}, H)$ to $\mathcal{A}$. In order to respond to Eval queries and the challenge X^*, $\mathcal{B}$ defines polynomials $\mathsf{P}_{X,i}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ – these polynomial will assume the role of the Θ_i's in the construction – for all $X \in \{0,1\}^*$ and $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$ as

$$\mathsf{P}_{X,i}(\mathsf{Z}) := \begin{cases} \mathsf{P}_{X,i-1}(\mathsf{Z}) \left(\tilde{w}_i \mathsf{Z} - K_i + H_i(X)\right) & \text{if } K_i \neq \perp \\ \mathsf{P}_{X,i-1}(\mathsf{Z}) \left(\tilde{w}_i + (H_i(X)\right) & \text{if } K_i = \perp \end{cases},$$

where $\mathsf{P}_{X,-1}(\mathsf{Z}) := 1$. Furthermore, we define $\mathsf{P}_X(\mathsf{Z}) := \mathsf{P}_{X,\lfloor \log(2\lambda+3) \rfloor}(\mathsf{Z})$. As in the proof of Theorem 3, we require Lemma 12. We observe that Lemma 12 implies that for all $0 \leq i \leq \lfloor \log(2\lambda + 3) \rfloor$ it holds that $\mathsf{Q}(\mathsf{Z})/\mathsf{P}_{X,i}(\mathsf{Z}) = \mathsf{R}_{X,i}(\mathsf{Z})$ for some polynomial $R_{X,i} \in \mathbb{Z}_p[\mathsf{Z}]$ if there exists $i \in \{0, \ldots, \ell\}$ such that $H_i(X) \neq K_i$, because we have $\mathsf{P}_{X,i} \mid \mathsf{P}_X(\mathsf{Z})$ by the definitions of $\mathsf{P}_X(\mathsf{Z})$ and $\mathsf{P}_{X,i}(\mathsf{Z})$.

Answering Eval queries. When $\mathcal{B}$ receives an Eval query $X \in \{0,1\}^*$ from $\mathcal{A}$, it checks whether $H_i(X) = K_i$ for all $i \in \mathcal{I}$ and if so aborts and outputs a random bit. If there is an index $i \in \mathcal{I}$ such that $H_i(X) \neq K_i$, let $\mathsf{R}_{X,i}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{R}_{X,i}(\mathsf{Z}) = \mathsf{Q}(\mathsf{Z})/\mathsf{P}_{X,i}(\mathsf{Z})$ for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$. Recall that such a polynomial $\mathsf{R}_{X,i}(\mathsf{Z})$ is guaranteed to exist by Lemma 12. $\mathcal{B}$ then computes the coefficients $\rho_{X,i,k} \in \mathbb{Z}_p$ of the polynomials $\mathsf{R}_{X,i}(\mathsf{Z})$ for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$, such that $\mathsf{R}_{X,i}(\mathsf{Z}) = \sum_{k=0}^{q} \mathsf{Z}^k \rho_{X,i,k}$ for all $i = 0, \ldots, \lfloor \log(2\lambda + 3) \rfloor$. $\mathcal{B}$ then computes π_i as

$$\prod_{k=0}^{q} [y\alpha^k]^{\rho_{X,i,k}} = [y \sum_{k=0}^{q} \alpha^k \rho_{X,i,k}] = [y\mathsf{R}_{X,i}(\alpha)] = [y\mathsf{Q}(\alpha)/\mathsf{P}_{X,i}(\alpha)] = [1/\mathsf{P}_{X,i}(\alpha)]$$

for all $0 \leq i \leq \lfloor \log(2\lambda + 3) \rfloor$ and $Y := e(\pi_{\lfloor \log(2\lambda+3) \rfloor}, [1]) = e([1], [1])^{1/\mathsf{P}_X(\alpha)}$. It then returns the result $(Y, \pi_0, \ldots, \pi_{\lfloor \log(2\lambda+3) \rfloor})$ to $\mathcal{A}$.

Answering challenge. When $\mathcal{B}$ receives the challenge $X^* \in \{0,1\}^*$, it checks if there is an index $i \in \mathcal{I}$ such that $H_i(X^*) \neq K_i$ and if so aborts and outputs a random bit. Otherwise, let $\zeta := \zeta_{X^*} \in \mathbb{Z}_p$ and $\mathsf{R}(\mathsf{Z}) := \mathsf{R}_{X^*}(\mathsf{Z}) = \sum_{k=0}^{q} \mathsf{Z}^k \gamma_k \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{Q}(\mathsf{Z})/\mathsf{P}_{X^*}(\mathsf{Z}) = \zeta_{X^*}/\mathsf{Z} + \mathsf{R}_{X^*}(\mathsf{Z})$ as guaranteed by Lemma 12. $\mathcal{B}$ then computes coefficients $\gamma_k \in \mathbb{Z}_p$ of R. Using that $\mathsf{Q}(\mathsf{Z}) = \sum_{k=0}^{q-1} \varphi_k \mathsf{Z}^k$, it also computes

$$Y = V^{\zeta \cdot \varphi_0} e\left([y], \prod_{k=1}^{q-1} [y\alpha^{k-1}]^{\zeta\varphi_k}\right) e\left(\prod_{k=0}^{q} [y\alpha^k]^{\gamma_k}, \prod_{k=0}^{q-1} [y\alpha^k]^{\varphi_k}\right) \tag{17}$$

and returns Y to $\mathcal{A}$. Finally, when $\mathcal{A}$ outputs its guess b' to $\mathcal{B}$, then $\mathcal{B}$ also outputs b' as solution to the q-DBDHI instance.

Analysis of $\mathcal{B}$. Recapping the construction of $\mathcal{B}$, we observe that vk and all answers of $\mathcal{B}$ are distributed identically to the challenger's interactions with $\mathcal{A}$ in Game 4. Analyzing $\mathcal{B}$'s response to the challenge X^*, we distinguish the following two cases depending on the q-DBDHI instance.
Let $V = e([y],[y])^{1/\alpha}$. Then we have for the first part of (17)

$$\begin{aligned} V^{\zeta\cdot\varphi_0} e\left([y], \prod_{k=1}^{q-1} [y\alpha^{k-1}]^{\zeta\varphi_k}\right) &= e([y],[y])^{\zeta\cdot\varphi_0/\alpha} e\left([y], \prod_{k=1}^{q-1} [y\alpha^{k-1}]^{\zeta\varphi_k}\right) \\ = e\left([y], \left[y\zeta \cdot \left(\frac{\varphi_0}{\alpha} + \sum_{k=1}^{q-1} \varphi_k \alpha^{k-1}\right)\right]\right) &= e\left([y],[y]\right)^{\zeta\mathsf{Q}(\alpha)/\alpha} \end{aligned}$$

and for the second part

$$e\left(\prod_{k=0}^{q} [y\alpha^k]^{\gamma_k}, \prod_{k=0}^{q-1} [y\alpha^k]^{\varphi_k}\right) = e\left([y\mathsf{R}(\alpha)], [y\mathsf{Q}(\alpha)]\right) = e\left([y],[y]\right)^{\mathsf{R}(\alpha)\mathsf{Q}(\alpha)}.$$

Together we have that

$$\begin{aligned} Y &= e\left([y],[y]\right)^{\zeta\cdot\mathsf{Q}(\alpha)/\alpha + \mathsf{R}(\alpha)\cdot\mathsf{Q}(\alpha)} = e\left([y],[y]\right)^{\mathsf{Q}(\alpha)^2/P_{X^*}(\alpha)} = e\left([1],[1]\right)^{1/P_{X^*}(\alpha)} \\ &= e\left([1],[1]\right)^{1/\Theta_{\lfloor\log(2\lambda+3)\rfloor}(X^*)} \end{aligned}$$

where we use Lemma 12 multiplied by $\mathsf{Q}(\mathsf{Z})$ on both sides, $[y\mathsf{Q}(\alpha)] = [1]$, and $P_{X^*}(\alpha) = \Theta_{\lfloor\log(2\lambda+3)\rfloor}(X^*)$
.
In case that $V \xleftarrow{\$} \mathbb{G}_T$ we have that Y is uniformly random in $\mathbb{G}_T$.
We observe that Y is distributed exactly as if the challenger answers with the VRF output in Game 4 in the first case and in the second case as if the challenger answers the challenge with a random element. Hence, $\mathcal{B}$ perfectly simulates Game 4 towards $\mathcal{A}$ and we have $\mathsf{Adv}_{\mathcal{B}}^{q\text{-}\mathsf{DBDHI}}(\lambda) = E_4$. We finish the proof of Theorem 4 by plugging together.

$$\mathsf{Adv}_{\mathcal{B}}^{q\text{-}\mathsf{DBDHI}}(\lambda) = E_4 \geq \varepsilon_{\mathcal{A}}^2/(32t_{\mathcal{A}}^2 - 16t_{\mathcal{A}}) - \mathsf{negl}(\lambda).$$

Running time of $\mathcal{B}$. Furthermore, the running time $t_{\mathcal{B}}$ of $\mathcal{B}$ consists of the running time $t_{\mathcal{A}}$ of $\mathcal{A}$ plus the time required to respond to evaluate queries by $\mathcal{A}$. Where, for the latter part, the most time consuming operations are the q exponentiations. Analogously to the proof of Theorem 3 we get

$$t_{\mathcal{B}} = O(t_{\mathcal{A}}^2/\varepsilon_{\mathcal{A}}),$$

which completes the proof of Theorem 4. □