J. Cryptol. (2014) 27: 544–593
DOI: 10.1007/s00145-013-9153-x

Journal of
CRYPTOLOGY

Verifiable Random Functions: Relations to Identity-Based Key Encapsulation and New Constructions*

Michel Abdalla

Département d'Informatique, Ecole Normale Supérieure, Paris, France
michel.abdalla@ens.fr

Dario Catalano

Dipartimento di Matematica e Informatica, Università di Catania, Catania, Italy
catalano@dmi.unict.it

Dario Fiore

Max Planck Institute for Software Systems (MPI-SWS), Saarbrücken, Germany
fiore@mpi-sws.org

Communicated by Nigel P. Smart

Received 4 January 2010
Online publication 25 May 2013

Abstract. In this paper we show a relation between the notions of verifiable random functions (VRFs) and identity-based key encapsulation mechanisms (IB-KEMs). In particular, we propose a class of IB-KEMs that we call VRF-suitable, and we propose a direct construction of VRFs from VRF-suitable IB-KEMs. Informally, an IB-KEM is VRF-suitable if it provides what we call *unique decapsulation* (i.e., given a ciphertext C produced with respect to an identity ID, all the secret keys corresponding to identity ID', decapsulate to the same value, even if $ID \neq ID'$), and it satisfies an additional property that we call *pseudo-random decapsulation*. In a nutshell, pseudo-random decapsulation means that if one decapsulates a ciphertext C, produced with respect to an identity ID, using the decryption key corresponding to any other identity ID', the resulting value looks random to a polynomially bounded observer. Our construction is of interest both from a theoretical and a practical perspective. Indeed, apart from establishing a connection between two seemingly unrelated primitives, our methodology is *direct* in the sense that, in contrast to most previous constructions, it avoids the inefficient Goldreich–Levin hardcore bit transformation. As an additional contribution, we propose a new VRF-suitable IB-KEM based on the decisional ℓ-weak Bilinear Diffie–Hellman Inversion assumption. Interestingly, when applying our transformation to this scheme, we obtain a new VRF construction that is secure under the same assumption, and it efficiently supports a large input space.

Key words. Verifiable random functions, Identity-based encryption, Pseudo-randomness.

* An abridged version of this paper appeared in the proceedings of EUROCRYPT 2009.

1. Introduction

Verifiable Random Functions (VRFs for short) were introduced by Micali, Rabin and Vadhan [41]. Informally, a VRF behaves like a pseudo-random function but also allows for efficient verification. More precisely, this means that there is a public key pk and a function F associated with a secret key sk (the seed) such that the following properties are satisfied. First, the function is efficiently computable, given sk, on any input. Second, having only pk and oracle access to the function, the value $F_{pk}(x) = y$ looks random to any polynomially bounded observer who did not query $F_{pk}(x)$ explicitly. Third, a proof $\pi_{pk}(x)$ that $F_{pk}(x) = y$ is efficiently computable knowing sk and efficiently verifiable knowing only pk.

VRFs turn out to be very useful in a variety of applications essentially because they can be seen as a compact commitment to an exponential number of (pseudo)random bits. To give a few examples, Micali and Reyzin [39] showed how to use VRFs to reduce to 3 the number of rounds of resettable zero-knowledge proofs in the bare model. Micali and Rivest [40] described a very simple non-interactive lottery system used in micro-payment schemes, based on VRFs. Jarecki and Shmatikov [34] employed VRFs to build a verifiable transaction escrow scheme that preserves users' anonymity while enabling automatic de-escrow. Liskov [36] used VRFs to construct updatable zero-knowledge databases.

However, in spite of their popularity, VRFs are not very well understood objects. In fact, only four constructions were known, in the standard model [20,22,38,41]. The schemes proposed by Micali, Rabin and Kilian [41] and by Lysyanskaya [38] build VRFs in two steps. First they focus on constructing a *Verifiable Unpredictable Function* (VUF), and then they show how to convert a VUF into a VRF using the Goldreich–Levin [28] theorem to "extract" random bits. Informally, a VUF is a function that is hard to compute but whose produced outputs do not necessarily look random. Unfortunately, the VRF resulting from this transformation is very inefficient and, furthermore, it loses a quite large factor in its exact security reduction. This is because the transformation involves several steps, all rather inefficient. First, one uses the Goldreich–Levin theorem [28] to construct a VRF with very small (i.e., slightly super polynomial in the security parameter) input space and output size 1. Next, one iterates the previous step in order to amplify the output size to (roughly) that of the input. Then, using a tree based construction, one iterates the resulting function in order to get a VRF with unrestricted input size and finally one evaluates the so obtained VRF several times in order to get an output size of the required length.

The constructions proposed by Dodis [20] and by Dodis and Yampolskiy [22], on the other hand, are direct, i.e., they manage to construct VRFs without having to resort to the Goldreich–Levin transform. The VRF presented in [20] is based on a "DDH-like" assumption that the author calls *sum-free decisional Diffie–Hellman* (sf-DDH). This assumption is similar to the one employed by Naor–Reingold [42] to construct PRFs, with the difference that it applies an error correcting code C to the input elements in order to compute the function. The specific properties of the employed encoding allow for producing additional values that can be used as proofs. This construction is more efficient than [38,41] in the sense that it does not need the expensive Goldreich–Levin transform. Still, it has some efficiency issues as the size of the produced proofs and keys is linear

in the input size. Dodis [20] also adapts this construction to provide a *distributed* VRF, that is a standard VRF which can be computed in a distributed manner. The scheme proposed by Dodis and Yampolskiy [22], on the other hand, is more attractive, at least from a practical point of view, as it provides a simple implementation of VRFs with short (i.e., constant-size) proofs and keys. It is interesting to note that, even though the latter construction is far more efficient than previous work, it builds upon a similar approach: first, they consider a simple VUF (which is basically the Boneh–Boyen weakly secure signature scheme [6]) that is secure for slightly superpolynomially sized input spaces, and then, rather than resorting to the Godreich-Levin [28] hardcore bit theorem to convert it into a VRF, they show how to modify the original VUF in order to make it a VRF, under an appropriate decisional assumption.

From the discussion above it seems clear that, with the possible exception of [20], all known constructions of verifiable random functions follow similar design criteria. First one builds a suitable VUF and then one transforms it into a VRF by either using the Goldreich–Levin transform or via some direct, ad hoc, modifications of the original VUF. The main drawback of this approach is that, once a good enough VUF is found, one has to either be able to convert it into a VRF directly or accept the fact that the VRF obtained via the Goldreich–Levin transform is not going to be a practical one.

The main motivating question of our work is whether there are alternative (and potentially more efficient) ways for constructing VRFs directly, without the need to resort to the two-step methodology sketched above.

1.1. *Our Contribution*

In this paper we show how to construct VRFs from a class of identity-based encryption (IBE) schemes [44] that we call *VRF-suitable*. In particular, we deal with the related notion of identity-based key encapsulation mechanisms (IB-KEM). Roughly speaking, an identity-based key encapsulation mechanism is an asymmetric encryption scheme where the public key can be an arbitrary string. Such schemes consist of four algorithms: a Setup algorithm that generates the system common parameters as well as a master key msk; a key derivation algorithm that uses the master secret key to generate a private key d_{sk} corresponding to an arbitrary public key string ID (the identity); an encapsulation algorithm that creates a ciphertext and a session key using the public key ID, and a decapsulation algorithm that recovers the session key from a ciphertext using the corresponding private key.

Informally, an IB-KEM is said to be VRF-suitable if the following conditions are met. First, the scheme has to provide *unique decapsulation*. This means that, given a ciphertext C produced with respect to some arbitrary identity ID, all the secret keys corresponding to any other identity ID' decapsulate to the same value (i.e., even if $ID' \neq ID$). Second, the IB-KEM has to provide what we call *pseudo-random decapsulation*. Very informally, pseudo-random decapsulation means that if C is a ciphertext produced using some identity ID, then the "decapsulated" key should look random even if the decapsulation algorithm is executed using the secret key corresponding to any other identity $ID^* \neq ID$. Having a scheme that achieves pseudo-random decapsulation may seem like a strong requirement at first. We argue that this is not the case by showing that several existing IBE schemes *already* provide pseudo-random decapsulation.

Our generic construction is of interest both from a theoretical and a practical point of view. Indeed, apart from establishing a connection between two seemingly unrelated primitives, our method is direct, in the sense that it allows to build a VRF from a VRF-suitable IB-KEM without having to resort to the inefficient Goldreich–Levin transform. Moreover, our reduction is tight. This means that, once an efficient VRF-suitable IB-KEM is available, this leads to an equally efficient VRF, with no additional security loss. Furthermore, our construction immediately allows for efficient distributed VRFs as long as a distributed version of the underlying encryption scheme is available (which is the case for most schemes used in practice).

Realizing VRF-Suitable IB-KEMs As a second contribution of this paper, we investigate on the possibility of realizing VRF-suitable IB-KEMs. Toward this goal, we first describe a general, but limited, construction from a class of standard public key encryption schemes. The proposed implementation (that we call q-bounded VRF-suitable IB-KEM) is limited in the sense that the pseudo-random decapsulation property is guaranteed to hold only if a restricted number of key derivations is allowed. This results in a VRF where the number of proofs that can be produced is bounded by q. Implementing the underlying public key encryption scheme using specific cryptosystems (such as ElGamal [24] or the Linear encryption scheme of Boneh, Boyen and Shacham [8]) we obtain efficient constructions of q-bounded VRFs from various number-theoretic assumptions.

Next, we show how to construct a fully fledged VRF-suitable IB-KEM from the Sakai–Kasahara IB-KEM [43]. Interestingly, the resulting VRF turns out to be very similar to the Dodis–Yampolskiy VRF [22]. Finally, we propose a new implementation of a VRF-suitable IB-KEM inspired (but more efficient) by Lysyanskaya's VRF [38] (which in turn builds from the Naor–Reingold's PRF [42]). Unlike Lysyanskaya's construction, whose security relies on the interactive Many-DH assumption [38], our new scheme can be proven secure based on the intractability, in bilinear groups, of the (non-interactive) decisional ℓ-weak Bilinear Diffie–Hellman Inversion problem (decisional ℓ-wBDHI* for short) introduced by Boneh, Boyen and Goh [9]. Our scheme enjoys several interesting properties. First, even though the decisional ℓ-wBDHI* assumption is asymptotic in nature, the ℓ parameter does not need to be too large in order for our security proof to go through. This is because ℓ is only related to the size of the identities, but it is *not* related to the number of adversarial queries allowed in the security reduction (as opposed to most known proofs using asymptotic assumptions). More precisely, ℓ grows logarithmically with respect to the size of the identity space. This means that in practice it is enough to assume the decisional ℓ-wBDHI* assumption to hold only for reasonably small values of ℓ (such as $\ell = 160$ or $\ell = 256$). Second, if one is interested only in selective-security,[1] then our scheme can efficiently support an unbounded input space. Finally, we prove our VRF-suitable IB-KEM to be fully-secure so that the resulting VRF can efficiently support *large input spaces*. To this end, we show two different proofs. The first one uses the notion of admissible hash functions introduced by Boneh and Boyen [5], and it requires to slightly change the scheme in the sense that

[1] A VRF-suitable IB-KEM is selective-secure if it satisfies a relaxed pseudo-random decapsulation property where the adversary declares at the beginning the identity on which it wishes to be challenged.

identities have to first be hashed using an admissible hash function. The second proof, instead, works for the same scheme that we prove selective-secure, and it obtains full security by using a slightly different computational assumption (n-Decisional Diffie–Hellman Exponent [10]) and the artificial abort technique [46]. Having constructions for large input spaces was an important open problem in the context of VRFs that was first solved in the recent works of Hohenberger and Waters [33], and Boneh *et al.* [11].

IBEs and Digital Signatures Naor [7] pointed out that a fully-secure identity-based encryption scheme can be transformed into a secure signature scheme as follows. One sets the message space as the set I of valid identities of the IBE. To sign $m \in I$, one executes the key derivation algorithm on input m, and outputs d_{sk} as the signature. A signature on m is verified by encrypting a random M with respect to the identity m, and then by checking that decrypting the resulting ciphertext one gets back M. Thus if one considers an IBE with unique key derivation (i.e., where for each identity a single corresponding decryption key can be computed), the methodology sketched above leads to a *unique* signature (i.e., a digital signature scheme for which each message admits one single valid signature). Since unique signatures are, by definition, verifiable unpredictable functions, at first glance our construction might seem to (somewhat) follow from Naor's remark. We argue that this is not the case for two reasons. First, our construction does not require the underlying IB-KEM to have unique key derivation, but only to provide unique decapsulation. Clearly, the former property implies the latter, but there is no reason to exclude the possibility of constructing a scheme realizing unique decapsulation using a randomized key derivation procedure. Second, a crucial requirement for Naor's transformation to work is that the original IBE is fully-secure. A VRF-suitable IB-KEM, on the other hand, is required to be secure only in a much weaker sense (that we call *weak selective* ID security).

1.2. *Other Related Work*

As pointed out above, the notion of VRF is related to the notion of unique signatures. Unique signatures were introduced by Goldwasser and Ostrovsky [29] (they called them invariant signatures). The only known constructions of unique signatures in the plain model (i.e., without common parameters or random oracles) are due to Micali, Rabin and Vadhan [41], to Lysyanskaya [38] and to Boneh and Boyen [6]. In the common reference string model, Goldwasser and Ostrovsky [29] also showed that unique signatures require the same kind of assumptions needed to construct non-interactive zero-knowledge.

Dodis and Puniya in [21] addressed the problem of constructing Verifiable Random Permutations (VRPs) from Verifiable Random Functions. They defined VRPs as the verifiable analogy of pseudo-random permutations. In particular, they pointed out that the technique of Luby–Rackoff [37] (for constructing PRPs from PRFs) cannot be applied in this case. This is due to the fact that VRP proofs must reveal the VRF outputs and proofs of the intermediate rounds. In their paper they showed a construction in which a super-logarithmic number of executions of the Feistel transformation suffices to build a VRP.

Chase and Lysyanskaya [16] introduced the notion of simulatable VRF. Informally, a simulatable VRF is a VRF with the additional property that proofs can be simulated, i.e.,

a simulator can fake proofs showing that the value of $F_{sk}(x)$ is y for any y of its choice. Simulatable VRFs work in the common reference string model, and they can be used to provide a direct transformation from single-theorem non-interactive zero-knowledge to multi-theorem NIZK.

Two works [12,26] have recently investigated the possibility of realizing VRFs from general assumptions. Brakerski *et al.* [12] introduced the notion of Weak Verifiable Random Functions (wVRFs) that are defined like standard VRFs except that the pseudo-randomness property holds only for randomly selected inputs. Brakerski *et al.* proposed constructions of wVRFs from trapdoor permutations and number-theoretic assumptions (Bilinear Diffie–Hellman), and they showed that weak VRFs are essentially equivalent to efficient prover non-interactive zero-knowledge proof systems. They also showed a black-box separation between VRFs (both weak and standard) and one-way permutations. In a recent work, Fiore and Schröder [26] further investigated the minimal cryptographic assumptions needed to realize VRFs, and showed that VRFs cannot be reduced in a black-box way to the existence of trapdoor permutations.

1.3. *Publication Note and Organization*

An abridged version of this paper appeared in the proceedings of EUROCRYPT 2009 [1]. In this version, we give more precise and formal definitions of VRF-suitable IB-KEMs, we include complete proofs of security, and we provide additional results. Most notably, this version contains the construction of a VRF scheme which is proven fully-secure for large input spaces. Moreover, we define q-bounded VRFs and we propose constructions based on standard public key encryption schemes such as ElGamal and Linear Encryption.

Organization The paper is organized as follows. Section 2 introduces some basic notation, defines the relevant computational assumptions used in our constructions, and gives the definitions of IB-KEM and VRF-suitable IB-KEM. Section 3 describes our generic construction of verifiable random functions from VRF-suitable IB-KEMs. Section 4 introduces the notion of q-bounded VRFs and proposes a realization based on public-key encryption schemes. Section 5 presents two constructions of VRF-suitable IB-KEMs, one based on the Sakai–Kasahara IB-KEM [43] and a new one for which we prove selective- and full-security for large identity spaces. Section 6 recalls our contributions and discusses future directions. The Appendix contains a few more results of independent interest. In particular, Appendix A motivates the notion of pseudo-random decapsulation by showing that the IB-KEMs of Waters [46] and Boneh–Franklin [7] satisfy this notion; and Appendix C shows an alternative proof of full security for the new VRF-suitable IB-KEM in Section 5 based on the security proof for the Hohenberger–Waters VRF [33].

2. Preliminaries

Before presenting our results, we briefly recall some basic definitions. In what follows we will denote by k the security parameter. An algorithm $\mathcal{A}$ is called PPT if it is a probabilistic Turing machine whose running time is bounded by some polynomial in k.

Denote by $\mathbb{N}$ the set of natural numbers and by $\mathbb{R}^+$ the set of positive real numbers. We say that a function $\epsilon : \mathbb{N} \to \mathbb{R}^+$ is negligible if and only if for every polynomial $P(k)$ there exists a $k_0 \in \mathbb{N}$ such that for all $k > k_0$, $\epsilon(k) < 1/P(k)$. If A is a set, then $a \xleftarrow{\$} A$ indicates the process of selecting a at random and uniformly over A (which in particular assumes that A can be sampled efficiently). If $\mathcal{A}(\cdot)$ is a PPT algorithm, then $y \xleftarrow{\$} \mathcal{A}(x)$ indicates the process of running $\mathcal{A}$ on input x and assigning its output to y.

2.1. *Assumptions*

In this section we recall some number-theoretic hardness assumptions that will be used in our work. In the following assume $\mathbb{G}$ to be a cyclic multiplicative group of prime order p, where p is a k-bit long prime and g is a generator of $\mathbb{G}$.

2.1.1. *Decisional Diffie–Hellman Assumption*

The Decisional Diffie–Hellman assumption (DDH) is the decisional version of the Computational Diffie–Hellman problem (CDH) informally defined in [19]. Informally, the DDH problem in a group $\mathbb{G}$ of prime order p is defined as follows. Let $a, b \xleftarrow{\$} \mathbb{Z}_p$ be chosen at random. An adversary for the DDH problem is given as input (g, g^a, g^b, g^c) and it must output 0 if it believes that $c = ab$, or 1 if c is random and independent in $\mathbb{Z}_p$.

More formally, we define the advantage of an adversary $\mathcal{A}$ into deciding DDH in $\mathbb{G}$ as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DDH}}(k) = \left|\Pr\left[\mathcal{A}\left(g, g^a, g^b, g^{ab}\right) = 0\right] - \Pr\left[\mathcal{A}\left(g, g^a, g^b, g^c\right) = 0\right]\right|,$$

where the probability is taken over the random choices of $a, b, c \in \mathbb{Z}_p$ and the coin tosses of $\mathcal{A}$.

Definition 1 (DDH). The Decisional Diffie–Hellman (DDH) assumption holds in $\mathbb{G}$ if, for any PPT adversary $\mathcal{A}$, its advantage, $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DDH}}(k)$, is negligible in k.

2.1.2. *Decision Linear Assumption*

The Decision Linear assumption (DLin) was first proposed by Boneh *et al.* in [8]. The Decision Linear problem in a group $\mathbb{G}$ of prime order p can be defined as follows. Let $u, v, h \xleftarrow{\$} \mathbb{G}$ and $a, b \xleftarrow{\$} \mathbb{Z}_p$ be chosen at random. An adversary for the Decision Linear problem in $\mathbb{G}$ is given as input $(u, v, h, u^a, v^b, h^c) \in \mathbb{G}^6$ and it must output 0 if it believes that $c = a + b$ and 1 if c is random in $\mathbb{Z}_p$. More formally, we define the advantage of an adversary $\mathcal{A}$ into deciding the Decision Linear problem in $\mathbb{G}$ as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DLin}}(k) = \left|\Pr\left[\mathcal{A}\left(u, v, h, u^a, v^b, h^{a+b}\right) = 0\right] - \Pr\left[\mathcal{A}\left(u, v, h, u^a, v^b, h^c\right) = 0\right]\right|.$$

Definition 2 (DLin). The Decision Linear assumption holds in $\mathbb{G}$ if, for any PPT adversary $\mathcal{A}$, its advantage, $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DLin}}(k)$, is negligible in k.

2.1.3. *Decisional Bilinear Diffie–Hellman Assumption*

The Bilinear Diffie–Hellman assumption (BDH for short) was first introduced by Boneh and Franklin in [7]. Here we present its decisional version (DBDH) that was used in several other works (e.g. [46]).

The Decisional Bilinear Diffie–Hellman problem is defined as follows. Let $\mathbb{G}, \mathbb{G}_T$ be two groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and let g be a generator of $\mathbb{G}$. Let $a, b, c \xleftarrow{\$} \mathbb{Z}_p$ be randomly chosen. Let (g, g^a, g^b, g^c, Z) be the input of an adversary whose goal it to decide whether $Z = e(g, g)^{abc}$ or $Z = e(g, g)^z$ for a random independent $z \xleftarrow{\$} \mathbb{Z}_p$. More formally, we define the advantage of $\mathcal{A}$ into solving the DBDH problem as

$$\begin{aligned}\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DBDH}}(k) = \big|&\Pr\big[\mathcal{A}\big(g, g^a, g^b, g^c, e(g, g)^{abc}\big) = 0\big] \\ &- \Pr\big[\mathcal{A}\big(g, g^a, g^b, g^c, e(g, g)^z\big) = 0\big]\big|.\end{aligned}$$

Definition 3 (DBDH). The Decisional Bilinear Diffie–Hellman assumption holds in bilinear groups $\mathbb{G}, \mathbb{G}_T$ if, for any PPT adversary $\mathcal{A}$, its advantage, $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DBDH}}(k)$, is at most negligible in k.

2.1.4. *Decisional Bilinear Diffie–Hellman Inversion Assumption*

The q-decisional Bilinear Diffie–Hellman Inversion assumption (DBDHI for short) was first introduced by Boneh and Boyen in [4].

Let $\mathbb{G}, \mathbb{G}_T$ be two groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and let g be a generator of $\mathbb{G}$. In the DBDHI problem the adversary is given a tuple $(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)})$ together with a value Z, and it must decide whether $Z = e(g, g)^{1/x}$ or $Z = e(g, g)^z$, for randomly chosen $x, z \in \mathbb{Z}_p$. More formally, we define the advantage of an algorithm $\mathcal{A}$ into solving the DBDHI problem as

$$\begin{aligned}\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DBDHI}}(k) = \big|&\Pr\big[\mathcal{A}\big(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)}, e(g, g)^{1/x}\big) = 0\big] \\ &- \Pr\big[\mathcal{A}\big(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)}, e(g, g)^z\big) = 0\big]\big|.\end{aligned}$$

Definition 4 (DBDHI). The DBDHI assumption holds in bilinear groups $\mathbb{G}, \mathbb{G}_T$ if, for any PPT adversary $\mathcal{A}$ and for any q polynomial in k, we have that $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DBDHI}}$ is at most negligible in k.

2.1.5. *Decisional Weak ℓ-Bilinear Diffie–Hellman Inversion Assumption*

The Decisional weak ℓ-Bilinear Diffie–Hellman Inversion (ℓ-wBDHI*) assumption was introduced by Boneh, Boyen and Goh in [9].

Let $\mathbb{G}, \mathbb{G}_T$ be two groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and let g be a generator of $\mathbb{G}$. In the ℓ-wBDHI* problem the adversary is given as input a tuple $(g^c, g^b, g^{b^2}, \ldots, g^{b^\ell})$ together with a value Z, and it must decide whether $Z = e(g, g)^{b^{\ell+1}c}$ or $Z = e(g, g)^z$ for randomly chosen $b, c, z \in \mathbb{Z}_p^*$. Formally, we define

the advantage of an algorithm $\mathcal{A}$ into solving the decisional ℓ-wBDHI* as

$$\mathbf{Adv}_{\mathcal{A}}^{\text{wBDHI}^*}(k) = \left|\Pr\left[\mathcal{A}\left(g^c, g^b, g^{b^2}, \ldots, g^{b^\ell}, e(g,g)^{b^{\ell+1}c}\right) = 0\right]\right.$$
$$\left. - \Pr\left[\mathcal{A}\left(g^c, g^b, g^{b^2}, \ldots, g^{b^\ell}, e(g,g)^{z}\right) = 0\right]\right|,$$

where the probability is taken over the random choices of $b, c, z \in \mathbb{Z}_p^*$.

Definition 5 (ℓ-wBDHI*). We say that the decisional ℓ-wBDHI* assumption holds in bilinear groups $\mathbb{G}, \mathbb{G}_T$ if, for any ℓ polynomial in k, and for any PPT adversary $\mathcal{A}$, its advantage, $\mathbf{Adv}_{\mathcal{A}}^{\text{wBDHI}^*}(k)$, is negligible in k.

Remark 1. Cheon showed in [17] an attack against the Strong Diffie–Hellman assumption and its related problems (among which the DBDHI used to prove the security of the Dodis–Yampolskiy VRF). This attack reduces the security of a factor $\sqrt{q}$, and it applies to ℓ-wBDHI* as well (with a factor $\sqrt{\ell}$). However, as we will see in Sect. 5.2, for the sake of our construction we need to assume that the ℓ-wBDHI* assumption holds only for rather small values of ℓ (e.g., $\ell = 160$ or $\ell = 256$). This means that in our case the security loss is not as significant as in Dodis–Yampolskiy's.

Finally, we notice that the assumptions *DBDHI* and *wBDHI* are related in the sense that the former implies the latter, but the converse is not known (we defer the interested reader to [9] for further discussions on these assumptions).

2.2. *Verifiable Random Functions*

Verifiable Random Functions (VRFs for short) were introduced by Micali, Rabin and Vadhan [41]. Intuitively, a VRF behaves like a pseudo-random function, but also allows for proofs of correctness of its outputs. More formally, a VRF is a triplet of algorithms $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$ providing the following functionalities. The key generation algorithm Gen is a probabilistic algorithm that takes as input the security parameter and produces a couple of matching public and private keys (vpk, vsk). The deterministic algorithm Func, on input the secret key vsk and the input x, computes $(\mathsf{F}_{vsk}(x), \mathsf{Prove}_{vsk}(x))$, where $v = \mathsf{F}_{vsk}(x)$ is the value of the VRF, and $\pi = \mathsf{Prove}_{vsk}(x)$ is its proof of correctness. The verification algorithm V takes as input a tuple (vpk, x, v, π) and outputs a bit indicating whether or not π is a valid proof that $\mathsf{F}_{vsk}(x) = v$.

Let $a : \mathbb{N} \to \mathbb{N} \cup \{*\}$ and $b : \mathbb{N} \to \mathbb{N}$ be functions computable in polynomial time (in k). Moreover, we assume that $a(k)$ and $b(k)$ are bounded by a polynomial in k, except if a takes the value $*$ (in this case we simply assume that the VRF can take inputs of arbitrary length). Let $\mathcal{D}$ and $\mathcal{R}$ be two sets of size $2^{a(k)}$ and $2^{b(k)}$ respectively. Formally, we say that $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$ is a VRF with input space $\mathcal{D}$ and output space $\mathcal{R}$, if the following conditions are met.

Domain Range Correctness: For all $x \in \mathcal{D}$ it has to be the case that $\mathsf{F}_{vsk}(x) \in \mathcal{R}$. We require this condition to hold with overwhelming probability (over the choices of (vpk, vsk)).

Provability: For all $x \in \mathcal{D}$ if $\mathsf{Prove}_{vsk}(x) = \pi$ and $\mathsf{F}_{vsk}(x) = v$ then $\mathsf{V}(vpk, x, v, \pi) = 1$. We require this condition to hold with overwhelming probability (over the choices of (vpk, vsk) and the coin tosses of V).

Uniqueness: No values $(vpk, x, v_1, v_2, \pi_1, \pi_2)$, such that $v_1 \neq v_2$, can satisfy (unless with negligible probability over the coin tosses of V) $\mathsf{V}(vpk, x, v_1, \pi_1) = \mathsf{V}(vpk, x, v_2, \pi_2) = 1$.

Pseudo-randomness: For all probabilistic polynomial time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we require that

$$\left| \Pr \left[b' = b \;\middle|\; \begin{array}{l} (vpk, vsk) \xleftarrow{\$} \mathsf{Gen}(1^k); (x, \omega) \leftarrow \mathcal{A}_1^{\mathsf{Func}(\cdot)}(vpk) \\ b \xleftarrow{\$} \{0,1\}; v_0 \leftarrow \mathsf{F}_{vsk}(x); v_1 \xleftarrow{\$} \mathcal{R} \\ b' \leftarrow \mathcal{A}_2^{\mathsf{Func}(\cdot)}(\omega, v_b) \end{array} \right] - \frac{1}{2} \right| \leq \epsilon(k)$$

where $\epsilon(k)$ is a negligible function. In the above experiment, the notation $\mathcal{A}^{\mathsf{Func}(\cdot)}$ indicates that $\mathcal{A}$ has oracle access to the algorithm Func. Also, in order to make this definition sensible, we impose that $\mathcal{A}$ cannot query the oracle on input x.

Roughly speaking, pseudo-randomness guarantees that the output of the function at any given point x, for which a proof has not been issued, looks random to any polynomially bounded observer.

Selective-Secure Verifiable Random Functions We introduce a new notion of VRF that we call *selective-secure VRF*. Informally speaking, a *selective-secure VRF* is a VRF with a relaxed pseudo-randomness property in which the adversary is required to commit ahead of time (i.e., before seeing the public key) to the input value it intends to attack. Sometimes, to point out the opposition with this selective notion, we will refer to VRFs that are secure in the usual sense as *fully-secure* VRFs.

More formally, a VRF is *selective-secure* if it satisfies the VRF definition given before except that the pseudo-randomness property is replaced by the following selective variant:

Selective Pseudo-randomness: For all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we require that

$$\left| \Pr \left[b' = b \;\middle|\; \begin{array}{l} (x, \omega) \leftarrow A_1(); (vpk, vsk) \xleftarrow{\$} \mathsf{Gen}(1^k) \\ b \xleftarrow{\$} \{0,1\}; v_0 \leftarrow \mathsf{F}_{vsk}(x); v_1 \xleftarrow{\$} \mathcal{R} \\ b' \leftarrow A_2^{\mathsf{Func}(\cdot)}(\omega, v_b) \end{array} \right] - \frac{1}{2} \right| \leq \epsilon(k)$$

where $\epsilon(k)$ is a negligible function, and the adversary cannot query x to the oracle $\mathsf{Func}(\cdot)$.

A straightforward reduction shows that any selective-VRF is also fully-secure at the price of a (significant) loss in the resulting security.

Proposition 1. *Let* VRF *be a VRF scheme with input space* $\mathcal{D}$ *of size* $2^{a(k)}$, *which is selective-secure with security* $\epsilon(k)$. *Then, the same scheme is also fully-secure with security* $\epsilon(k)/2^{a(k)}$.

Proof. The proof of this proposition can be obtained by considering the following reduction. Let $\mathcal{A}$ be an adversary that breaks the pseudo-randomness of VRF with advantage at least $\epsilon(k)$. Then, one can build an adversary $\mathcal{B}$ that simulates $\mathcal{A}$ while playing

the selective pseudo-randomness experiment. At the beginning, $\mathcal{B}$ chooses a random point $x^* \xleftarrow{\$} \mathcal{D}$ as its challenge value and gives it to its challenger. Observe that $\mathcal{B}$ can perfectly simulate $\mathcal{A}$, as long as $\mathcal{A}$ does not query the oracle on x^* and it does ask its challenge on x^*. Otherwise $\mathcal{B}$ aborts and outputs a random bit. Since $\mathcal{B}$'s guess on x^* is correct with probability $1/2^{a(k)}$, $\mathcal{B}$ will have advantage $\epsilon(k)/2^{a(k)}$ of winning in the selective pseudo-randomness experiment. □

Remark 2. According to the definition given above, the output of a VRF is in a specific set $\mathcal{R}$. However, in some applications one may need a VRF whose output is a binary string. To this end, we note that any VRF with output space $\mathcal{R}$ can be turned into one that outputs a binary string by applying a suitable universal hash function to the output. More precisely, let $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$ be a VRF with output space $\mathcal{R}$, and let $\mathcal{H} = \{H : \mathcal{R} \to \{0,1\}^b\}_H$ be a family of universal hash functions (for a suitable b). Then we define another VRF scheme $\mathsf{VRF}' = (\mathsf{Gen}', \mathsf{Func}', \mathsf{V}')$ whose output space is $\{0,1\}^b$ and that works as follows. The key generation Gen' runs $(vpk, vsk) \xleftarrow{\$} \mathsf{Gen}(1^k)$, chooses a function H from the family $\mathcal{H}$ uniformly at random, and outputs $vpk' = (vpk, H)$ and $vsk' = vsk$. The algorithm Func' first obtains the output y and the proof π by running Func, and then it returns $y' = H(y)$ as the VRF's output, while it includes y in the proof, i.e., $\pi' = (\pi, y)$. Finally, the verification algorithm V' checks that $y' = H(y)$, and that π is a correct proof for y (by running V). It is easy to see that, since H is a deterministic function and the scheme VRF has uniqueness, VRF' satisfies uniqueness as well. Moreover, by the leftover hash lemma [2,30], if H is a universal hash and VRF satisfies pseudo-randomness over $\mathcal{R}$, then VRF' satisfies pseudo-randomness over $\{0,1\}^b$. For the sake of precision, we notice that the secure re-use of the same universal hash H (which is randomly generated once and then fixed in the public key) is justified by a variant of the leftover hash lemma proven by Shoup in [45].

2.3. *Identity-Based Encryption and Identity-Based Key Encapsulation*

An identity-based encryption scheme IBE consists of a tuple of algorithms $(\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ providing the following functionality. The trusted authority runs Setup, on input the security parameter 1^k, to generate a master key pair (mpk, msk). Without loss of generality, we assume that the public key mpk specifies a message space $\mathcal{M}$ and a value n (polynomial in the security parameter) indicating the length of each identity. The trusted authority publishes the master public key mpk, and keeps the master secret key msk private. When a user with identity ID wishes to become part of the system, the trusted authority generates a user decryption key $sk_{ID} \xleftarrow{\$} \mathsf{KeyDer}(msk, ID)$ and sends this key over a secure and authenticated channel to the user. To send an encrypted message m to the user with identity ID, the sender computes a ciphertext $C \xleftarrow{\$} \mathsf{Enc}(mpk, ID, m)$, which can be decrypted by the user as $m \leftarrow \mathsf{Dec}(mpk, d_{ID}, C)$.

Boneh and Franklin [7] formally defined the notion of security for identity-based encryption schemes. In particular, they defined the notion of chosen plaintext security against adaptive chosen identity attacks. Intuitively, such a notion captures the requirement that security should be preserved even when facing an adversary who is allowed to choose the identity it wishes to attack and to collude with (i.e., to obtain the secret keys of) other identities of the system. Later, Canetti, Halevi, and Katz [13] introduced

a weaker notion of security in which the adversary has to commit ahead of time (i.e., before the parameters of the scheme are made public) to the identity it intends to attack. A scheme meeting such a weaker security requirement is called selective-ID chosen plaintext secure IBE (IND-sID-CPA, for short).

Identity-Based Key Encapsulation In our work we will not use directly the notion of IBE, but we will rather consider the closely related notion of identity-based key encapsulation (IB-KEM). Therefore, we will provide formal definitions only for IB-KEMs.

An identity-based key encapsulation mechanism (IB-KEM) scheme enables a sender and a receiver to agree on a session key K in such a way that the sender can create K from public parameters and receiver's identity while the receiver can recover K using his secret key. This notion, in the context of identity-based encryption, was first formalized by Bentahar et al. [3].

More formally, an IB-KEM scheme is defined by the following four algorithms:

- $\mathsf{Setup}(1^k)$ is a probabilistic algorithm that takes as input a security parameter k, and outputs a master public key mpk and a master secret key msk. The master public key implicitly defines the identity space $\mathcal{ID}$ and the session key space $\mathcal{K}$.
- $\mathsf{KeyDer}(msk, ID)$ is the key derivation algorithm that uses the master secret key to compute a secret key sk_{ID} for an identity $ID \in \mathcal{ID}$.
- $\mathsf{Encap}(mpk, ID)$ is the encapsulation algorithm that computes a random session key K and a corresponding ciphertext C encrypted under the identity ID.
- $\mathsf{Decap}(mpk, ID, sk_{ID}, C)$ is the decapsulation algorithm that allows the possessor of a secret key sk_{ID} for the identity ID to decapsulate a ciphertext C to get back a session key K.

For correctness, it is required that $\forall k \in \mathbb{N}$, for all possible $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}(1^k)$, $\forall ID \in \mathcal{ID}$, $(C, K) \xleftarrow{\$} \mathsf{Encap}(mpk, ID)$ the following probability holds:

$$\Pr\big[\mathsf{Decap}\big(mpk, ID, \mathsf{KeyDer}(msk, ID), C\big) = K\big] = 1.$$

Security The standard notion of security for IB-KEM is semantic security against adaptive chosen-ID attacks (IB-KEM-CPA) and is recalled below.

Let $\mathcal{IBKEM}$ be an IB-KEM scheme, and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a PPT algorithm. Then consider the following experiment between a challenger and an adversary $\mathcal{A}$:

Setup: The challenger runs the Setup algorithm, keeps the master secret key msk for himself, and runs $\mathcal{A}_1(mpk)$ on input the master public key.

Phase 1: The adversary $\mathcal{A}_1$ is allowed to ask an arbitrary (but polynomially limited) number of key derivation queries. In each of these queries the adversary specifies an identity ID of its choice, and gets back the corresponding private key (which is generated by the challenger by running the algorithm $\mathsf{KeyDer}(msk, ID)$). The queries may be asked adaptively, i.e., each query can depend on previously issued ones.

Challenge: When Phase 1 is over, the adversary $\mathcal{A}_1$ outputs a state information st and an identity ID^* such that ID^* was not queried during Phase 1. The challenger then computes $(C, K_0) \xleftarrow{\$} \mathsf{Enc}(mpk, ID^*)$ and chooses a random session key $K_1 \xleftarrow{\$} \mathcal{K}$. Next, it picks a random bit b and runs the adversary $\mathcal{A}_2$ on input (st, C, K_b).

Phase 2: This phase goes exactly as Phase 1 with the additional restriction that $\mathcal{A}_2$ cannot issue a key derivation query for identity ID^*.

Guess: When Phase 2 is over, $\mathcal{A}_2$ outputs a bit b' denoting its guess for the bit b.

We define the advantage of an adversary $\mathcal{A}$ in attacking the scheme $\mathcal{IBKEM}$ as

$$\mathbf{Adv}^{\text{IB-KEM-CPA}}_{\mathcal{IBKEM},\mathcal{A}}(k) = \left|\Pr\left[b = b'\right] - \frac{1}{2}\right|$$

where the probability is taken over the internal coin tosses of the challenger and the adversary.

SELECTIVE-ID SECURITY. The notion of selective-identity security for IB-KEMs is defined similarly to the notion of IB-KEM-CPA security except that $\mathcal{A}_1$ is required to choose the identity ID^* before seeing the master public key. More precisely, the corresponding security game works as follows:

Initialize: $\mathcal{A}_1$ is run on input the security parameter and outputs an identity ID^* and a state information st.

Setup: The challenger runs the Setup algorithm and keeps the master secret key msk for himself. Next, it computes $(C, K_0) \xleftarrow{\$} \mathsf{Enc}(mpk, ID^*)$, and chooses a random session key $K_1 \xleftarrow{\$} \mathcal{K}$. It picks a random bit b and finally runs the adversary $\mathcal{A}_2$ on input (st, mpk, C, K_b).

Key derivation queries: The adversary $\mathcal{A}_2$ is allowed to ask an arbitrary (but polynomially limited) number of key derivation queries. In each of these queries the adversary specifies an identity ID of its choice such that $ID \neq ID^*$. $\mathcal{A}_2$ receives back the corresponding private key (which is generated by the challenger by running the algorithm $\mathsf{KeyDer}(msk, ID)$). The queries may be asked adaptively, meaning with this that each query can depend on previously issued ones.

Guess: At the end of the experiment, $\mathcal{A}_2$ outputs a bit b' denoting its guess for the bit b.

We define the advantage of an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the selective security of the scheme $\mathcal{IBKEM}$ as

$$\mathbf{Adv}^{\text{sIB-KEM-CPA}}_{\mathcal{IBKEM},\mathcal{A}}(k) = \left|\Pr\left[b = b'\right] - \frac{1}{2}\right|$$

where the probability is taken over the internal coin tosses of the challenger and the adversary.

WEAK SELECTIVE-ID SECURITY. In this paper we additionally introduce a new notion of security for IB-KEM schemes that we call *weak selective-ID security*. More precisely, we define this notion as the full fledged selective case with the exception that here the challenge identity is chosen by the challenger and given as input to the adversary. Clearly, this notion is weaker with respect to selective-ID security, and it is easy to see that the latter implies the former.

Formally, let us consider an efficiently samplable distribution $\mathcal{D}_{\mathcal{ID}}$ over the identity space,[2] and let $\mathcal{A}$ be a PPT adversary. We define the notion of *weak selective-ID security* (wsIB-KEM-CPA) for IB-KEM schemes by considering the following game:

[2] A distribution $\mathcal{D}$ is efficiently samplable if there exists a PPT algorithm that can sample elements according to $\mathcal{D}$.

Setup: In this phase the challenger selects a challenge identity $ID^* \leftarrow \mathcal{D}_{\mathcal{ID}}$ (according to distribution $\mathcal{D}_{\mathcal{ID}}$), and runs $(mpk, msk) \leftarrow \mathsf{Setup}(1^k)$. Then it computes $(C, K_0) = \mathsf{Encap}(mpk, ID^*)$ and chooses a random key $\bar{K} \xleftarrow{\$} \mathcal{K}$. Finally, it flips a binary coin $b \xleftarrow{\$} \{0, 1\}$, and runs $\mathcal{A}$ on input (mpk, ID^*, C, K_b).

Key derivation queries: The adversary is allowed to ask key derivation queries for an arbitrary (but polynomial) number of adaptively chosen identities different from ID^*.

Guess: In the end of this game $\mathcal{A}$ outputs b' as its guess for b.

We define the advantage of $\mathcal{A}$ against $\mathcal{IBKEM}$ in the above game w.r.t. distribution $\mathcal{D}_{\mathcal{ID}}$ as

$$\mathbf{Adv}^{\text{wsIB-KEM-CPA}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k) = \left|\Pr\left[b = b'\right] - \frac{1}{2}\right|$$

where the probability is taken over the coin tosses of the challenger and the adversary.

2.4. *VRF-Suitable IB-KEMs*

Our VRF construction relies on a special class of identity-based key encapsulation mechanisms that we call *VRF-suitable*. A VRF-suitable IB-KEM is defined by the following algorithms:

- $\mathsf{Setup}(1^k)$ is a probabilistic algorithm that takes as input a security parameter k and outputs a master public key mpk and a master secret key msk. We denote by $\mathcal{K}$ the session key space.
- $\mathsf{KeyDer}(msk, ID)$ is the key derivation algorithm that uses the master secret key to compute a secret key sk_{ID} for identity ID and some auxiliary information aux_{ID} needed to correctly encapsulate and decapsulate the key.
- $\mathsf{Encap}(mpk, ID, aux_{ID})$ is the encapsulation algorithm that computes a ciphertext C and a session key K using (mpk, ID, aux_{ID}). More precisely, in the algorithm the ciphertext C can be computed using only (mpk, ID), while the computation of K requires aux_{ID} in addition to (mpk, ID).
- $\mathsf{Decap}(mpk, ID, sk_{ID}, aux_{ID}, C)$ is the decapsulation algorithm that allows the possessor of sk_{ID} and aux_{ID} to decapsulate C to get back a session key K.

Remark 3. Note that the description above differs from the one given for basic IB-KEM in that here we allow the encapsulation and decapsulation mechanisms to use some auxiliary information aux_{ID}, produced by KeyDer, to work correctly. Clearly, if one sets $aux_{ID} = \perp$, then one goes back to the original description. Thus, the new paradigm is slightly more general as it allows to consider encapsulation mechanisms where everybody can compute the ciphertext but only those knowing the information aux_{ID} can compute the key. Notice however, that aux_{ID} does not allow, by itself, to decapsulate. In some sense, this auxiliary information should be seen as a value that completes the public key, rather than something that completes the secret key. In fact, this auxiliary information is not required to be kept secret in our constructions. Specifically, in all notions of security for VRF-suitable IB-KEMs (e.g., pseudo-random decapsulation and weak selective-id security) the adversary is allowed to obtain the auxiliary information for any identity of its choice, including the challenge identity. Even though

such a syntax may look useless in the standard public key scenario, it turns out to be extremely useful (see below) in our context.

In order to be *VRF-suitable*, an IB-KEM has to satisfy the following properties:

1. **Unique Decapsulation**. No tuples $(mpk, C_0, ID, sk_{ID}, aux_{ID}, sk'_{ID}, aux'_{ID})$, such that $(sk_{ID}, aux_{ID}) \neq (sk'_{ID}, aux'_{ID})$ can satisfy the following checks (unless with negligible probability over the coin tosses of Encap and Decap):
   - (i) $\mathsf{Decap}(C_0, sk_{ID}, aux_{ID}) \neq \mathsf{Decap}(C_0, sk'_{ID}, aux'_{ID}) \neq \perp$, and
   - (ii) both the following checks hold: $(C, K) \xleftarrow{\$} \mathsf{Encap}(mpk, ID, aux_{ID})$ and $K = \mathsf{Decap}(C, sk_{ID}, aux_{ID})$, $(C', K') \xleftarrow{\$} \mathsf{Encap}(mpk, ID, aux'_{ID})$ and $K' = \mathsf{Decap}(C', sk'_{ID}, aux'_{ID})$.

   We remark that an IB-KEM with a deterministic key derivation algorithm does not necessarily satisfy unique decapsulation. Intuitively, to see this, think of the case in which the key derivation algorithm generates the randomness using a PRF whose seed is part of the master secret key. Then, by fixing the master public key, one can still have two different seeds that lead to two different secret keys for the same identity. Therefore, what we require in unique decapsulation is that, even though an identity may have different secret keys, they all decapsulate a ciphertext to the same session key.
2. **Pseudo-random Decapsulation.** Let C be an encapsulation produced using some identity $ID_0 \in \mathcal{ID}$. Informally, this property says that the session key obtained by decapsulating the ciphertext C should look random even if C is decapsulated by executing the decapsulation algorithm using a secret key corresponding to any other $\overline{ID} \neq ID_0$.
   More formally, let $\mathcal{IBKEM}$ be an IB-KEM, $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a PPT adversary and let $\mathcal{D}_{\mathcal{ID}}$ be an efficiently samplable distribution over the identity space $\mathcal{ID}$. We define the pseudo-random decapsulation experiment as follows:

   Experiment $\mathbf{Exp}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k)$
   Choose $ID_0 \in \mathcal{ID}$ (according to $\mathcal{D}_{\mathcal{ID}}$)
   $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}(1^k)$
   $C^* \xleftarrow{\$} \mathsf{Encap}(mpk, ID_0)$
   $(\overline{ID}, st) \xleftarrow{\$} \mathcal{A}_1^{\mathsf{KeyDer}(\cdot)}(mpk, C^*, ID_0)$
   $(aux_{\overline{ID}}, sk_{\overline{ID}}) \xleftarrow{\$} \mathsf{KeyDer}(msk, \overline{ID})$
   $b \xleftarrow{\$} \{0,1\}$; $K_0 \xleftarrow{\$} \mathsf{Decap}(mpk, \overline{ID}, sk_{\overline{ID}}, aux_{\overline{ID}}, C^*)$; $K_1 \xleftarrow{\$} \mathcal{K}$
   $b' \leftarrow \mathcal{A}_2^{\mathsf{KeyDer}(\cdot)}(st, K_b, aux_{\overline{ID}})$
   If $b' = b$ then return 1, else return 0

   With $\mathcal{A}^{\mathsf{KeyDer}(\cdot)}$ we denote that $\mathcal{A}$ has oracle access to the key derivation algorithm. Let $\mathcal{ID}$ denote the identity space, i.e., the space from which the adversary (and everybody else) is allowed to choose the identities. In order to make the experiment $\mathbf{Exp}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}$ non-trivial, we introduce the following restrictions:

– the identity $\overline{ID}$ output by $\mathcal{A}_1$ should not have been asked to the $\mathsf{KeyDer}(\cdot)$ oracle;
– $\mathcal{A}_2$ is not allowed to query the oracle $\mathsf{KeyDer}(\cdot)$ on $\overline{ID}$.

We define the advantage of $\mathcal{A}$ in the experiment IB-KEM-RDECAP for the distribution $\mathcal{D}_{\mathcal{ID}}$ as

$$\mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k) = \left|\Pr\left[\mathbf{Exp}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k) = 1\right] - \frac{1}{2}\right|.$$

Finally, we say that $\mathcal{IBKEM}$ has pseudo-random decapsulation with respect to $\mathcal{D}_{\mathcal{ID}}$ if, for any PPT adversary $\mathcal{A}$, its advantage $\mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k)$ is a negligible function in k. Also, we simply say that $\mathcal{IBKEM}$ has pseudo-random decapsulation if there exists a distribution $\mathcal{D}_{\mathcal{ID}}$ for which the above holds.

The above definition essentially rules out all those schemes in which the decapsulation algorithm returns $\perp$, or any other error message, when the identities associated with a given secret key and ciphertext do not match. In other words, a necessary condition for an IBE to be VRF-suitable is that its decapsulation procedure always outputs some random looking key, even when invoked with an "incorrect" secret key.

We also note that, although requiring an IB-KEM to provide pseudo-random decapsulation may seem like a strong requirement, we argue that this is not the case. Indeed, several existing IB-KEM schemes which are IND-CPA secure (but not IND-CCA secure) *already* have this property. In Appendix A.1 we prove that the IB-KEM derived from the Waters' IBE scheme [46] provides pseudo-random decapsulation, while in Sect. 5.1 we prove that the same holds for the IB-KEM by Sakai and Kasahara [43]. It is easy to generalize these two proofs to the case of Boneh–Boyen (BB1) [4] and to the schemes of Boneh–Boyen (BB2) [4] and Gentry [27], respectively. However, with the exception of the Sakai–Kasahara's scheme and our new scheme in Sect. 5.2, all these schemes do not satisfy the unique decapsulation property.

In the following theorem we show that a necessary condition, in order for an IB-KEM to be VRF-suitable, is that it is secure in a weak selective sense.

Theorem 1. *Let $\mathcal{IBKEM}$ be a VRF-suitable IB-KEM satisfying pseudo-random decapsulation for a distribution $\mathcal{D}_{\mathcal{ID}}$, then it is also a weak selective-ID secure IB-KEM w.r.t. $\mathcal{D}_{\mathcal{ID}}$ (in the sense of the definition given in Sect. 2.3).*

Proof. Assume, for the sake of contradiction, that there exists an adversary $\mathcal{A}$ that breaks the weak selective-ID security of the given VRF-suitable IB-KEM. We show how to use this adversary to construct another adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ that refutes the pseudo-random decapsulation of the scheme. $\mathcal{B}_1$ is run on input (mpk, ID_0, C) (where ID_0 is chosen according to $\mathcal{D}_{\mathcal{ID}}$) and proceeds as follows. First, $\mathcal{B}_1$ outputs $\overline{ID} = ID_0$ as the challenge identity. Next, $\mathcal{B}_2$ receives a session key K_b (which is either the right decapsulation key corresponding to C or a random one) and an auxiliary information aux_{ID_0}. So, $\mathcal{B}_2$ runs $\mathcal{A}$ on input $(mpk, C, K_b, ID_0, aux_{ID_0})$. Whenever $\mathcal{A}$ asks for a key derivation query, $\mathcal{B}_2$ uses its own oracle to answer such query, in the obvious way. Finally, when $\mathcal{A}$ outputs a bit b', $\mathcal{B}_2$ outputs the same b'. It is easy to see that the simulation

is perfect and thus the advantage of $\mathcal{B}$ in breaking the pseudo-random decapsulation is exactly the same as the advantage of $\mathcal{A}$ in breaking the weak selective security of the scheme. □

Selective Pseudo-random Decapsulation In what follows we define a selective variant of pseudo-random decapsulation in which the adversary commits ahead of time to the identity on which it wishes to be challenged. Let $\mathcal{D}_{\mathcal{ID}}$ be an efficiently samplable distribution over the identity space as defined before. The experiment for selective pseudo-random decapsulation is defined as follows.

Experiment $\mathbf{Exp}^{\text{IB-KEM-selRDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k)$

Choose $ID_0 \in \mathcal{ID}$ (according to $\mathcal{D}_{\mathcal{ID}}$)
$(\overline{ID}, st) \xleftarrow{\$} \mathcal{A}_1(1^k, ID_0)$
$(mpk, msk) \xleftarrow{\$} \mathsf{Setup}(1^k)$
$C^* \xleftarrow{\$} \mathsf{Encap}(mpk, ID_0)$
$(aux_{\overline{ID}}, sk_{\overline{ID}}) \xleftarrow{\$} \mathsf{KeyDer}(msk, \overline{ID})$
$b \xleftarrow{\$} \{0,1\}$; $K_0 \xleftarrow{\$} \mathsf{Decap}(mpk, \overline{ID}, sk_{\overline{ID}}, aux_{\overline{ID}}, C^*)$; $K_1 \xleftarrow{\$} \mathcal{K}$
$b' \leftarrow \mathcal{A}_2^{\mathsf{KeyDer}(\cdot)}(st, mpk, C^*, K_b, aux_{\overline{ID}})$
If $b' = b$ then return 1, else return 0

In this experiment the adversary is not allowed to query the oracle on $\overline{ID}$. Like in standard pseudo-random decapsulation, $\mathcal{A}$'s advantage in the experiment IB-KEM-selRDECAP is defined as

$$\mathbf{Adv}^{\text{IB-KEM-selRDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k) = \left| \Pr\left[\mathbf{Exp}^{\text{IB-KEM-selRDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k) = 1\right] - \frac{1}{2} \right|.$$

Finally, we say that $\mathcal{IBKEM}$ satisfies selective pseudo-random decapsulation w.r.t. $\mathcal{D}_{\mathcal{ID}}$ if, for any PPT adversary $\mathcal{A}$, the advantage $\mathbf{Adv}^{\text{IB-KEM-selRDECAP}}_{\mathcal{IBKEM},\mathcal{A},\mathcal{D}_{\mathcal{ID}}}(k)$ is a negligible function in k. We say that a VRF-suitable IB-KEM that satisfies selective pseudo-random decapsulation is *selective-secure*. Otherwise, we say that it is *fully-secure*.

An analogue of Proposition 1, it can also be proved for VRF-suitable IB-KEMs.

Proposition 2. *Let $\mathcal{IBKEM}$ be a VRF-suitable IB-KEM with identity space $\mathcal{ID}$ that is selective-secure with security $\epsilon(k)$. Then, the same scheme is also fully-secure with security $\epsilon(k)/|\mathcal{ID}|$.*

3. A Generic Construction

In this section we show our construction of Verifiable Random Functions from a VRF-suitable IB-KEM $\mathcal{IBKEM} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Encap}, \mathsf{Decap})$. Let $\mathcal{ID}$ be the identity space, $\mathcal{K}$ the session key space and $\mathcal{SK}$ the secret key space. Also, let $\mathcal{D}_{\mathcal{ID}}$ be the distribution for which $\mathcal{IBKEM}$ satisfies pseudo-random decapsulation. Then we construct a verifiable random function $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$ with input space $\mathcal{ID}$ and output space $\mathcal{K}$ as follows.

- $\mathsf{Gen}(1^k)$ runs $(mpk, msk) \leftarrow \mathsf{Setup}(1^k)$, chooses an identity $ID_0 \leftarrow \mathcal{ID}$ according to the distribution $\mathcal{D}_{\mathcal{ID}}$, and computes $C_0 \leftarrow \mathsf{Encap}(mpk, ID_0)$. Finally, it returns $vpk = (mpk, C_0, ID_0)$ and $vsk = msk$.
- $\mathsf{Func}_{vsk}(x)$ computes $\pi_x = (sk_x, aux_x) = \mathsf{KeyDer}(msk, x)$ and $y = \mathsf{Decap}(mpk, x, \pi_x, C_0)$. It returns (y, π_x) where y is the output of the function and π_x is the proof.
- $\mathsf{V}(vpk, x, y, \pi_x)$ first checks if π_x is a valid proof for x in the following way. It computes $(C, K) = \mathsf{Encap}(mpk, x, aux_x)$ and checks if $K = \mathsf{Decap}(mpk, \pi_x, C)$. Next, it checks the validity of y by testing if $\mathsf{Decap}(mpk, x, \pi_x, C_0) = y$. If both the tests are true, then the algorithm returns 1, otherwise it returns 0.

3.1. *Security Proof*

Now we prove that the proposed construction actually realizes a secure VRF.

Theorem 2. *Assume $\mathcal{IBKEM}$ is a VRF-suitable IB-KEM scheme, as described in Sect. 2, then the construction given above is a verifiable random function.*

Proof. According to the definition given in Sect. 2, we prove that $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Func}, \mathsf{V})$ is a verifiable random function by showing that it satisfies all the properties. Domain range correctness and provability trivially follow from the correctness of the IB-KEM scheme.

To see that the uniqueness property is satisfied, we show that it is implied by the unique decapsulation of $\mathcal{IBKEM}$. Recall that the latter property says that there cannot exist a tuple $(mpk, C_0, ID, sk_{ID}, aux_{ID}, sk'_{ID}, aux'_{ID})$ such that $(sk_{ID}, aux_{ID}) \neq (sk'_{ID}, aux'_{ID})$, and that satisfies:

(i) $\mathsf{Decap}(mpk, ID, sk_{ID}, aux_{ID}, C_0) \neq \mathsf{Decap}(mpk, ID, sk'_{ID}, aux'_{ID}, C_0) \neq \perp$, and

(ii) $(C, K) \xleftarrow{\$} \mathsf{Encap}(mpk, ID, aux_{ID})$ and $K \leftarrow \mathsf{Decap}(mpk, ID, sk_{ID}, aux_{ID}, C)$, $(C', K') \xleftarrow{\$} \mathsf{Encap}(mpk, ID, aux'_{ID})$ and $K' \leftarrow \mathsf{Decap}(mpk, ID, sk'_{ID}, aux'_{ID}, C')$.

By construction, this can be restated as requiring that there cannot exist a tuple (vpk, x, π_x, π'_x) where $\pi_x \neq \pi'_x$, and it is such that: (i) $y = \mathsf{Decap}(mpk, x, sk_x, aux_x, C_0) \neq \mathsf{Decap}(mpk, x, sk'_x, aux'_x, C_0) = y'$, and (ii) $(C, K) \xleftarrow{\$} \mathsf{Encap}(mpk, x, aux_x)$ and $K \leftarrow \mathsf{Decap}(mpk, x, sk_x, aux_x, C)$, $(C', K') \xleftarrow{\$} \mathsf{Encap}(mpk, x, aux'_x)$ and $K' \leftarrow \mathsf{Decap}(mpk, x, sk'_x, aux'_x, C')$. Observe that all the checks can be equivalently rewritten as: $y \neq y'$, $y = \mathsf{Decap}(mpk, x, sk_x, aux_x, C_0)$, $(C, K) \xleftarrow{\$} \mathsf{Encap}(mpk, x, aux_x)$ and $K \leftarrow \mathsf{Decap}(mpk, x, sk_x, aux_x, C)$, $y' = \mathsf{Decap}(mpk, x, sk'_x, aux'_x, C_0)$, $(C', K') \xleftarrow{\$} \mathsf{Encap}(mpk, x, aux'_x)$ and $K' \leftarrow \mathsf{Decap}(mpk, x, sk'_x, aux'_x, C')$. Again, by our definition of the verification algorithm, this is in turn equivalent to saying that $y \neq y'$ and $\mathsf{V}(vpk, x, y, \pi_x) = \mathsf{V}(vpk, x, y', \pi'_x) = 1$, that is the uniqueness of VRFs.

To prove pseudo-randomness, we assume by contradiction that there exists a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that is able to break the pseudo-randomness of VRF with non-negligible advantage $\epsilon(k)$. Then we show how to build a PPT adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ which uses $\mathcal{A}$ to obtain non-negligible advantage $\epsilon(k)$ in the IB-KEM-RDECAP experiment for distribution $\mathcal{D}_{\mathcal{ID}}$.

$\mathcal{B}_1$ receives from its challenger a public key mpk, a ciphertext C_0^*, and an identity ID_0 chosen according to $\mathcal{D}_{\mathcal{ID}}$. So, $\mathcal{B}_1$ sets $vpk = (mpk, C_0^*, ID_0)$ and runs $\mathcal{A}_1(vpk)$. The adversary $\mathcal{A}$ is allowed to make queries to the function oracle $\mathsf{Func}(\cdot)$, and $\mathcal{B}$ simulates this oracle as follows. On input a value $x \in \mathcal{ID}$, $\mathcal{B}$ queries its key derivation oracle on x, it obtains (sk_x, aux_x) and returns $(y_x = \mathsf{Decap}(mpk, x, sk_x, aux_x, C_0^*), \pi_x = (sk_x, aux_x))$ to the adversary. When $\mathcal{A}_1$ outputs an element $\bar{x}$, $\mathcal{B}_1$ outputs the same element to its challenger. Thus the challenger produces K^*, which is either the decapsulation of C_0^* with $(sk_{\bar{x}}, aux_{\bar{x}})$ or a random element of $\mathcal{K}$, and gives K^* to $\mathcal{B}_2$. Finally, $\mathcal{B}_2$ runs $b' \leftarrow \mathcal{A}_2(st, K^*)$ (simulating all oracle queries as $\mathcal{B}_1$) and returns the bit b' to its challenger.

It is easy to see that $\mathcal{B}$ is perfectly simulating the pseudo-randomness game to $\mathcal{A}$. Thus, if $\mathcal{A}$ has advantage $\epsilon(k)$, then $\mathcal{B}$'s advantage is exactly the same. □

4. q-Bounded VRFs

In the previous section we described a general transformation that allows to construct verifiable random functions from a class of identity-based key encapsulation mechanisms that we call VRF-suitable. Before showing, in Sect. 5, two VRF-suitable IB-KEMs that lead to two VRFs, in this section we introduce a slightly weaker notion of verifiable random functions that we call *q-bounded VRFs*. A q-bounded VRF is a standard VRF (as defined in Sect. 2.2) with the limitation that pseudo-randomness is preserved only if at most q proofs are produced.

4.1. *Construction of q-Bounded VRFs from Public Key Encryption Schemes*

We show that it is possible to construct a q-bounded VRF from a public key encryption scheme. This construction consists of two steps. First, a q-resilient IB-KEM (which is defined below) is constructed from any IND-CPA secure encryption scheme. Later we can apply our generic transformation to build a q-bounded VRF from a q-resilient IB-KEM that is VRF-suitable.

Building Blocks Before describing the construction in detail, we discuss some preliminary building blocks.

4.1.1. *Key Encapsulation Mechanism*

We briefly describe the notion of public key encryption schemes with key encapsulation mechanism (KEM for short) and its related definition of security.

A KEM is defined by three algorithms:

- $\mathsf{Kg}(1^k)$ is the key generation algorithm that takes as input the security parameter k and outputs a pair of keys (pk, sk) where pk is made public and sk is kept secret.
- $\mathsf{Encap}(pk)$ is a probabilistic algorithm that takes as input the public key pk and outputs (C, K), where C is the ciphertext and $K \in \mathcal{K}$ is a session key.
- $\mathsf{Decap}(pk, sk, C)$ is a deterministic algorithm that takes as input the public key pk, the secret key sk, and a ciphertext C, and it outputs either a key K or an error symbol $\perp$.

We note that, given any standard public key encryption scheme, it is always possible to construct a KEM. We define the notion of indistinguishability under chosen-plaintext attacks (KEM-IND-CPA) for KEMs. Essentially, it is the usual IND-CPA security notion for public key encryption, adapted to the KEM setting. Consider the following experiment:

Experiment $\mathbf{Exp}^{\text{KEM-IND-CPA}}_{\text{KEM},\mathcal{A}}(k)$

$(pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k)$

$(C_0, K_0) \xleftarrow{\$} \mathsf{Encap}(pk)$, $K_1 \xleftarrow{\$} \mathcal{K}$

$b \xleftarrow{\$} \{0, 1\}$

$b' \leftarrow \mathcal{A}(pk, C_0, K_b)$

If $b' = b$ then return 1, else return 0

The advantage of $\mathcal{A}$ in the experiment above is defined as

$$\mathbf{Adv}^{\text{KEM-IND-CPA}}_{\text{KEM},\mathcal{A}}(k) = \left|\Pr\left[\mathbf{Exp}^{\text{KEM-IND-CPA}}_{\text{KEM},\mathcal{A}}(k) = 1\right] - 1/2\right|.$$

We say that a KEM is KEM-IND-CPA-secure if any PPT adversary $\mathcal{A}$ has at most negligible advantage in the experiment $\mathbf{Exp}^{\text{KEM-IND-CPA}}_{\text{KEM},\mathcal{A}}$.

Unique decapsulation for KEM: We define the notion of unique decapsulation for KEMs. A KEM satisfies unique decapsulation if there are no tuples (pk, sk, sk', C_0) such that $sk \neq sk'$ and $\mathsf{Decap}(pk, sk, C) \neq \mathsf{Decap}(pk, sk', C) \neq \perp$.

Pseudo-random decapsulation for KEM: Here we introduce the notion of pseudo-random decapsulation for public key encryption schemes with key encapsulation. Consider the following experiment:

Experiment $\mathbf{Exp}^{\text{KEM-RDECAP}}_{\text{KEM},\mathcal{A}}(k)$

$(pk_0, sk_0) \xleftarrow{\$} \mathsf{Kg}(1^k)$

$(pk_1, sk_1) \xleftarrow{\$} \mathsf{Kg}(1^k)$

$(C^*, K^*) \xleftarrow{\$} \mathsf{Encap}(pk_0)$

$b \xleftarrow{\$} \{0, 1\}$

$K_0 \xleftarrow{\$} \mathsf{Decap}(pk_1, sk_1, C^*)$

$K_1 \xleftarrow{\$} \mathcal{K}$

$b' \leftarrow \mathcal{A}(pk_0, sk_0, pk_1, C^*, K_b)$

If $b' = b$ then return 1, else return 0

The advantage of $\mathcal{A}$ in the experiment above is defined as

$$\mathbf{Adv}^{\text{KEM-RDECAP}}_{\text{KEM},\mathcal{A}}(k) = \left|\Pr\left[\mathbf{Exp}^{\text{KEM-RDECAP}}_{\text{KEM},\mathcal{A}}(k) = 1\right] - 1/2\right|.$$

We say that a KEM satisfies pseudo-random decapsulation if any PPT adversary $\mathcal{A}$ has at most negligible advantage in the experiment $\mathbf{Exp}^{\text{KEM-RDECAP}}_{\text{KEM},\mathcal{A}}$.

q-Resilient IB-KEMs In [31] Heng and Kurosawa introduced the notion of q-resilient security in the context of identity-based encryption [31]. Informally, an IB-KEM is said to be q-resilient if it is secure only when facing adversaries that are allowed to issue at most q key derivation queries. Later, Cramer *et al.* [18] pointed out that, this notion, was implicitly given by Dodis, Katz, Xu, and Yung in [23], when introducing the notion of key-insulated public-key cryptosystems.

Cover-Free Families If S, T are sets, we say that S does not cover T if $S \not\supseteq T$. Let d, q, s be positive integers, and let $F = (F_i)_{1 \le i \le s}$ be a family of subsets of $\{1, \ldots, d\}$. We say that family F is *q-cover-free over* $\{1, \ldots, d\}$, if for each subset $F_i \in F$ and each S that is the union of at most q sets in $(F_1, \ldots, F_{i-1}, F_{i+1}, \ldots, F_s)$, it is the case that S does not cover F_i. Furthermore, we say that F is l-uniform if all subsets in the family have size l. We use the following fact [25,35]: there is a deterministic polynomial time algorithm that on input integers s, q returns l, d, F where $F = (F_i)_{1 \le i \le s}$ is an l-uniform q-cover-free family over $\{1, \ldots, d\}$, for $l = d/4q$ and $d \le 16q^2 \log(s)$. In the following we let SUB denote the resulting deterministic polynomial-time algorithm that on input s, q, i returns F_i. We call $F_i = \mathsf{SUB}(s(k), q(k), i)$ the subset associated with index $i \in \{1, \ldots, s(k)\}$.

For our construction we will need a cover-free family with parameters

$$s(k) = 2^k, \quad d(k) = 16kq^2(k), \quad l(k) = 4kq(k). \tag{1}$$

4.1.2. *q-Resilient IB-KEM from KEM*

Here we show how to construct a q-resilient IB-KEM from a KEM. Such construction is given in [18,31]. Here we adapt it to encompass the KEM case.

Let $q(k), d(k), s(k), l(k) : \mathbb{N} \to \mathbb{N}$ be (efficiently computable) functions. For ease of exposition we simply refer to them as q, d, s, l. Let $\mathcal{KEM} = (\mathsf{Kg}, \mathsf{Encap}, \mathsf{Decap})$ be a public key encryption scheme with key encapsulation and let F be an l-uniform *q-cover-free* family over $\{1, \ldots, d(k)\}$. We denote by $F_{ID} = \mathsf{SUB}(s(k), l(k), ID) = \{r_1, \ldots, r_l\}$ the subset associated with an identity ID. We assume identities are integers in $\{1, \ldots, s(k)\}$.

We construct the IB-KEM $\mathcal{IBKEM} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Encap}, \mathsf{Decap})$ in the following way:

- $\mathsf{Setup}(1^k, q)$: For $i = 1, \ldots, d$ compute $(pk_i, sk_i) \leftarrow \mathsf{Kg}(1^k)$. Set $mpk = (pk_1, \ldots, pk_d)$ and $msk = (sk_1, \ldots, sk_d)$.
- $\mathsf{KeyDer}(msk, ID)$: Given $F_{ID} = \{r_1, \ldots, r_l\}$ (recall that $F_{ID} = \mathsf{SUB}(s(k), l(k), ID) = \{r_1, \ldots, r_l\}$), set $SK_{ID} = (sk_{r_1}, \ldots, sk_{r_l})$.
- $\mathsf{Encap}(mpk, ID)$: Let $F_{ID} = \{r_1, \ldots, r_l\}$ and compute $(c_i, K_i) = \mathsf{Encap}(pk_i)$ for $i = r_1, \ldots, r_l$. Set $C = (c_{r_1}, \ldots, c_{r_l})$ and $K = K_{r_1} \oplus \cdots \oplus K_{r_l}$.
- $\mathsf{Decap}(mpk, ID, SK_{ID}, C)$: Let $SK_{ID} = (sk_{r_1}, \ldots, sk_{r_l})$. Compute $K_i = \mathsf{Decap}(pk_i, sk_i, c_i)$ for $i = r_1, \ldots, r_l$ and set $K = K_{r_1} \oplus \cdots \oplus K_{r_l}$.

Theorem 3. *If $\mathcal{KEM}$ is a* KEM-IND-CPA*-secure key encapsulation mechanism and F is a q-cover-free family over $\{1, \ldots, d\}$, then the scheme $\mathcal{IBKEM}$ described above is* (q*-resilient*) IB-KEM-CPA*-secure.*

Proof. For the sake of contradiction, assume there exists an efficient adversary $\mathcal{A}$ that is able to break the IB-KEM-CPA security of $\mathcal{IBKEM}$ with non-negligible probability $\epsilon(k)$. Then we show how to build an efficient algorithm $\mathcal{B}$ that can break the KEM-IND-CPA security of the underlying KEM scheme with advantage at least $\epsilon(k)/d(k)$, where $d(k)$ is the parameter for the cover-free family as described above.

First observe that there exists an index $j \in \{1, \ldots, d\}$ such that j belongs to the subset $F_{\overline{ID}}$ associated with the challenge identity $\overline{ID}$ but not to any subset associated with the identities queried to the key derivation oracle. As long as the number of key derivation queries is at most q, we know such an index must exist. Moreover, as d is of polynomial size, such index j can be guessed by our algorithm $\mathcal{B}$ with non-negligible probability $1/d$.

$\mathcal{B}$ takes as input a public key pk and constructs the master public key of the IB-KEM as follows. It generates $(pk_i, sk_i) \xleftarrow{\$} \mathsf{Kg}(1^k)$ $\forall i = 1, \ldots, j-1, j+1, \ldots, d$, sets $pk_j = pk$ and gives $mpk = (pk_1, \ldots, pk_d)$ to $\mathcal{A}$. Now observe that if the guess of j is right, then $\mathcal{B}$ is able to answer all key derivation queries made by $\mathcal{A}$ (as $\mathcal{B}$ knows all the secret keys but sk_j). At some point the adversary supplies a challenge identity $\overline{ID}$ such that $F_{\overline{ID}} = \{r_1, \ldots, r_\ell\}$ contains j and $\mathcal{B}$ answers as follows. It computes $k_i \xleftarrow{\$} \mathsf{Encap}_{pk_i}()$ $\forall i \in F_{\overline{ID}} \setminus \{j\}$, then queries its encapsulation oracle, gets back $\bar{k}$ and sets $k_j = \bar{k}$. $\mathcal{B}$ sets $\overline{K} = k_{r_1} \oplus \cdots \oplus k_{r_l}$ and gives it to $\mathcal{A}$. When $\mathcal{A}$ outputs its decision bit b, the algorithm $\mathcal{B}$ outputs the same bit as its guess about the distribution of $\bar{k}$.

It is easy to see that when the guess of j is right the simulation provided by $\mathcal{B}$ is perfect. Indeed if $\mathcal{B}$ is given a random $\bar{k}$ then $\overline{K}$ is also random, otherwise $\overline{K}$ is a correctly distributed session key. Thus $\mathcal{B}$ wins with probability at least ϵ/d. □

q-Bounded VRFs from q-Resilient IB-KEMs Recall that our final goal is a construction of q-bounded VRFs from public key encryption schemes (with specific properties). The first step is to show how to construct a q-bounded VRF from a q-resilient IB-KEM. However, this can be obtained very easily by applying our generic transformation given in Sect. 3 to a q-resilient VRF-suitable IB-KEM.

Therefore, to obtain the final result the only thing we have to ensure is that the q-resilient IB-KEM obtained through the construction given in the previous section is VRF-suitable, that is it satisfies pseudo-random decapsulation and unique decapsulation.

First, to see that the q-resilient IB-KEM has unique decapsulation, we observe that this holds if the underlying KEM satisfies the analogous unique decapsulation (for KEMs). Indeed, assume for the sake of contradiction there is a tuple $(mpk, C_0, ID, sk_{ID}, sk'_{ID})$ such that $sk_{ID} \neq sk'_{ID}$ and both conditions (i) and (ii) hold. Since the choice of the public/secret keys related to ID is a deterministic and public process, notice that condition (i) already implies that there is (at least) a tuple (pk, sk, sk', c_0) such that $sk \neq sk'$ and $\mathsf{Decap}(pk, sk, c_0) \neq \mathsf{Decap}(pk, sk', c_0) \neq \perp$. Hence, if the q-resilient IB-KEM does not have unique decapsulation, so is for the underlying KEM.

Second, we formally prove in the following theorem that the resulting scheme has pseudo-random decapsulation if the original KEM, on top of being KEM-IND-CPA secure, satisfies the analogous (for KEM) pseudo-random decapsulation property defined

before. As for the case of IBEs, this might seem a quite strong requirement at first. However, as we will show in Sect. 4.2, it is not too hard to find examples of public key encryption schemes whose KEM version already achieves pseudo-random decapsulation.

Theorem 4. *If a KEM satisfies pseudo-random decapsulation then the resulting q-resilient IB-KEM obtained via the transformation given in Sect. 4.1.2 has pseudo-random decapsulation as well.*

Proof. In particular, our theorem shows that our q-resilient IB-KEM satisfies pseudo-random decapsulation for any arbitrary distribution $\mathcal{D}_{\mathcal{ID}}$.

Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for the pseudo-random decapsulation of the q-resilient IB-KEM and, for the sake of contradiction, assume that $\mathcal{A}$ has non negligible advantage $\mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{A},\mathcal{IBKEM}}(k) = \epsilon(k)$. Then we show how to build a simulator $\mathcal{B}$ that can break either the KEM-IND-CPA security or the pseudo-random decapsulation of the underlying KEM with non negligible advantage.

Let ID_0 be an identity chosen by the simulator according to $\mathcal{D}_{\mathcal{ID}}$, and let $\overline{ID}$ be the challenge identity returned by the adversary. We distinguish two cases:

1. $\overline{ID} = ID_0$
2. $\overline{ID} \neq ID_0$

$\mathcal{A}$ will output a challenge identity either of type 1 or type 2 with probability at least $1/2$. We will show a simulator $\mathcal{B}$ that in the first case breaks the KEM-IND-CPA security of the KEM, whereas in the second case it breaks the pseudo-random decapsulation of the KEM.

At the beginning, $\mathcal{B}$ flips a binary coin $\beta \xleftarrow{\$} \{0, 1\}$ and runs Simulation β as described below. Basically, if $\beta = 0$ $\mathcal{B}$ guesses that $\mathcal{A}$ will output a challenge identity of type 1, whereas, if $\beta = 1$, $\mathcal{B}$ guesses that $\mathcal{A}$ will output a challenge identity $\overline{ID} \neq ID_0$. We stress that these simulations are perfectly indistinguishable from the adversary's point of view.

SIMULATION 0. In this case $\mathcal{B}$ acts as an adversary for the KEM-IND-CPA security of the KEM. It receives in input (pk, C^*, K^*). Since $\mathcal{B}$ is guessing that $\overline{ID} = ID_0$, this proof is the same as that one for the wsIB-KEM-CPA security of the IB-KEM. Let $F_{ID_0} = \{r_1, \ldots, r_l\}$ be the subset associated with the identity ID_0. $\mathcal{B}$ picks a random index $j \xleftarrow{\$} F_{ID_0}$ and sets $pk_{r_j} = pk$. Then it generates the remaining $d - 1$ pairs of keys $(pk_i, sk_i) = \mathsf{Kg}()$ $\forall i = 1, \ldots, d$ and $i \neq j$ and sets $mpk = (pk_1, \ldots, pk_d)$. It computes $(C_i, K_i) = \mathsf{Encap}(pk_i) \forall i \neq j$ and sets the ciphertext as $C_0 = (C_1, \ldots, C_{j-1}, C^*, C_{j+1}, \ldots, C_l)$. It runs $\mathcal{A}_1(mpk, C_0)$. $\mathcal{A}_1$ issues key derivation queries until it outputs the challenge identity $\overline{ID}$. For every key derivation query ID asked by the adversary, let F_{ID} be its associated subset. If $j \in F_{ID}$, then $\mathcal{B}$ aborts and outputs a random bit $b \in \{0, 1\}$. Otherwise, it uses the secret keys to compute the key of the identity ID.

Let $\overline{ID}$ be the challenge identity returned by $\mathcal{A}_1$. If $\overline{ID} \neq ID_0$ $\mathcal{B}$ aborts. Otherwise, let $F_{ID_0} = \{s_1, \ldots, s_l\}$ be the subset associated with identity ID_0. $\mathcal{B}$ sets $K_{s_j} = K^*$, $K = K_{s_1} \oplus \cdots \oplus K_{s_l}$ and runs $b' \leftarrow \mathcal{A}_2(K)$. Finally, $\mathcal{B}$ outputs the same bit b'.

We observe that if K^* is a random session key, so is K. Otherwise if $K^* = \mathsf{Decap}(pk_{i^*}, sk_{i^*}, C^*)$ then K is properly distributed. In this case let us consider the

probability that the simulator wins when $\mathcal{A}$ wins and $\mathcal{A}$ outputs $\overline{ID} = ID_0$ as challenge identity. This is equal to the probability that $\mathcal{A}$ wins and $\mathcal{B}$ does not abort in the key derivation phase. Since the adversary issues at most q queries, we know that there exists at least an index j such that j is not in any of the subsets associated with the queried identities and $j \in F_{\overline{ID}}$. Since $\overline{ID} = ID_0$, $\mathcal{B}$ does not abort in the key derivation phase with probability at least $1/l$ (independent of $\mathcal{A}$'s view).

SIMULATION 1. In this case $\mathcal{B}$ acts as an adversary for the pseudo-random decapsulation of the KEM. It receives in input (pk, sk, pk', C^*, K^*). Let $F_{ID_0} = \{r_1, \ldots, r_l\}$ be the subset associated with the identity ID_0. $\mathcal{B}$ picks a random index $j \xleftarrow{\$} \{1, \ldots, l\}$ and sets $pk_{r_j} = pk$ and $sk_{r_j} = sk$. Then it picks another index $i^* \xleftarrow{\$} \{1, \ldots, d\}$ and sets $pk_{i^*} = pk'$. Later it generates the remaining $d - 2$ pairs of keys $(pk_i, sk_i) = \mathsf{Kg}()\ \forall i = 1, \ldots, d$ and $i \neq r_j, i^*$ and sets $mpk = (pk_1, \ldots, pk_d)$. It computes $(C_i, K_i) = \mathsf{Encap}(pk_i)\ \forall i \neq j$ and constructs the ciphertext as $C_0 = (C_1, \ldots, C_{j-1}, C^*, C_{j+1}, \ldots, C_l)$. It runs $\mathcal{A}_1(mpk, C_0)$. $\mathcal{A}_1$ issues key derivation queries until it outputs the challenge identity $\overline{ID}$. Let ID be a queried identity and let F_{ID} be its associated subset. If $\overline{ID} = ID_0$ or $i^* \in F_{ID}$ then $\mathcal{B}$ aborts and outputs a random bit $b \in \{0, 1\}$. Otherwise it uses the known secret keys to compute the key of the identity ID.

Let $F_{\overline{ID}} = \{s_1, \ldots, s_l\}$ be the subset associated with the challenge identity $\overline{ID}$. If $i^* \notin F_{\overline{ID}}$ $\mathcal{B}$ aborts and outputs a random bit. Otherwise let j' be the index such that $s_{j'} = i^*$. If $j' \neq j$ $\mathcal{B}$ aborts. Otherwise it sets $K_{s_j} = K^*$, $K = K_{s_1} \oplus \cdots \oplus K_{s_l}$ and runs $b' \leftarrow \mathcal{A}_2(K)$. Then $\mathcal{B}$ outputs the same b'.

We observe that if K^* is a random session key, so is K. Otherwise if $K^* = \mathsf{Decap}(pk_{i^*}, sk_{i^*}, C^*)$ then K is properly distributed. In this simulation we have two abort conditions:

1. $i^* \notin F_{\overline{ID}}$ or i^* is in a subset associated with one of the identities queried to the key derivation oracle;
2. $j \neq j'$.

The first abort condition does not happen with probability at least $1/d$, since we know that, as long as at most q queries are asked, there exists an index $i \in \{1, \ldots, d\}$ such that i belongs to $F_{\overline{ID}}$ and not to any of the subsets associated with the identities queried to the key derivation oracle. The second abort condition does not happen with probability $1/l$. Thus $\mathcal{B}$ does not abort with probability at least $1/dl$ (independent of $\mathcal{A}$'s view). In this case $\mathcal{B}$ wins when $\mathcal{A}$ wins and $\mathcal{A}$ outputs $\overline{ID} \neq ID_0$, and $\mathcal{B}$ does not abort.

Let we denote by fail the event that the simulator fails. Thus in both the simulations $\mathcal{B}$ wins with advantage $\epsilon(k) \cdot \Pr[\overline{\mathsf{fail}}]$. In conclusion, we have that the simulator breaks either the KEM-IND-CPA security of the KEM with advantage at least $\epsilon(k)/2l$, or it breaks the pseudo-random decapsulation of the KEM with advantage at least $\epsilon(k)/(2dl)$. □

4.2. *Practical Examples*

In this section we show two examples of PKE schemes that satisfy the pseudo-random decapsulation and unique decapsulation properties (and thus they can be used to construct q-bounded VRFs). The first is the well known ElGamal encryption scheme [24], while the second one is the Linear Encryption scheme by Boneh, Boyen and Shacham [8].

q-Bounded VRFs from ElGamal Here we prove that the KEM version of the ElGamal encryption scheme satisfies the unique decapsulation and pseudo-random decapsulation properties defined in Sect. 4.1.1. First we recall the scheme:

- $\mathsf{Kg}(1^k)$: Let $\mathbb{G}$ be a group of order p, and $g \in \mathbb{G}$ be a generator. The key generation algorithm picks a random $x \xleftarrow{\$} \mathbb{Z}_p$. It sets the public key $pk = (\mathbb{G}, g, X = g^x)$ and the secret key $sk = x$.
- $\mathsf{Encap}(pk)$: The encapsulation algorithm picks a random $y \xleftarrow{\$} \mathbb{Z}_p$ and produces a ciphertext $C = g^y$ and a session key $K = X^y$.
- $\mathsf{Decap}(pk, sk, C)$: The decapsulation algorithm first checks that $X = g^{sk}$. If this does not hold, output $\perp$. Otherwise, it uses the secret key to "extract" a session key from a given ciphertext C by computing $K = C^x$.

It is easy to see that the scheme satisfies unique decapsulation because of the check in the Decap algorithm and the fact that for each public key, there is only one secret key (moreover, this is efficiently checkable).

So, we are left with proving pseudo-random decapsulation in the following theorem.

Theorem 5. *The KEM version of the ElGamal encryption scheme satisfies pseudo-random decapsulation under the Decisional Diffie–Hellman assumption.*

Proof. Let $\mathcal{A}$ be an adversary for the pseudo-random decapsulation of the scheme above. Then we show how to construct a simulator $\mathcal{B}$ that exploits $\mathcal{A}$ to break the Decisional Diffie–Hellman (DDH) assumption (see Sect. 2.1.1).

$\mathcal{B}$ receives in input a DDH tuple (g, g^a, g^b, Z). It picks random $x_0 \xleftarrow{\$} \mathbb{Z}_p$ and sets $pk_0 = (\mathbb{G}, g, g^{x_0})$, $sk_0 = x_0$, $pk_1 = (\mathbb{G}, g, g^a)$, $C^* = g^b$, $K = Z$. Then it runs $\mathcal{A}$ on input $(pk_0, sk_0, pk_1, C^*, K)$ and gets back a bit b'. In the end the simulator outputs b'.

If Z is g^{ab} then K is correctly distributed. Indeed we have $K = (C^*)^{sk_1} = \mathsf{Decap}(pk_1, sk_1, C^*)$. Otherwise if Z is random K is random too. Thus the simulation is perfect and $\mathcal{B}$ achieves the same advantage of $\mathcal{A}$. □

q-Bounded VRFs from Linear Encryption With an argument similar to that of ElGamal, it can be easily proved that also the Linear Encryption scheme described in [8] by Boneh, Boyen and Shacham has pseudo-random decapsulation under the so-called Decision Linear Assumption (see Sect. 2.1.2).

First, we recall the scheme. Let G be a group of prime order p.

- $\mathsf{Kg}(1^k)$: The key generation algorithm selects three elements $u, v, h \xleftarrow{\$} \mathbb{G}$ uniformly at random and computes $x, y \in \mathbb{Z}_p$ such that $u^x = v^y = h$. The public key is $pk = (u, v, h)$ while the secret key is $sk = (x, y)$.
- $\mathsf{Encap}(pk)$: The encapsulation algorithm picks random $a, b \xleftarrow{\$} \mathbb{Z}_p$ and returns the ciphertext $C = (u^a, v^b)$ and the session key $K = h^{a+b}$.
- $\mathsf{Decap}(pk, sk, C)$: The decapsulation algorithm first checks that $h = u^x = v^y$. If this does not hold, then it returns $\perp$. Otherwise, it uses the secret key sk to compute the session key associated with a given ciphertext $C = (C_1, C_2)$ by computing $K = C_1^x C_2^y$.

Similarly to the case of ElGamal, the scheme has unique decapsulation because of the check in the Decap algorithm and the fact that for each public key there is only one secret key, and this is efficiently checkable.

Theorem 6. *The KEM version of the Linear Encryption scheme satisfies pseudo-random decapsulation under the Decision Linear assumption.*

Proof. Let $\mathcal{A}$ be an adversary that has non-negligible advantage into breaking the pseudo-random decapsulation of the Linear Encryption scheme given above. Then we show how to build an efficient simulator $\mathcal{B}$ that solves the Decision Linear problem with non-negligible probability.

$\mathcal{B}$ receives in input a tuple (u, v, h, u^a, v^b, Z). It picks random $h_0 \xleftarrow{\$} G$, $x_0, y_0 \xleftarrow{\$} \mathbb{Z}_p$ and sets $u_0 = h_0^{1/x_0}$, $v_0 = h_0^{1/y_0}$. It sets $pk_0 = (u_0, v_0, h_0)$, $sk_0 = (x_0, y_0)$, $pk_1 = (u, v, h)$, $C^* = (u^a, v^b)$ and $K = Z$. Since there exist $x, y \in \mathbb{Z}_p$ such that $u^x = v^y = h$, $\mathcal{B}$ is implicitly setting $sk_1 = (x, y)$. Then the simulator runs $b \leftarrow \mathcal{A}(pk_0, sk_0, pk_1, C^*, K)$ and outputs the same b.

We show that the simulation is perfect and thus $\mathcal{B}$ wins with the same probability $\mathcal{A}$ wins.

First of all, observe that there exist $\tau_u, \tau_v \in \mathbb{Z}_p$ such that $u_0 = u^{\tau_u}$ and $v_0 = v^{\tau_v}$. Thus C^* is a valid ciphertext for pk_0, i.e. it can be written as $C^* = (u_0^{a'}, v_0^{b'})$ where $a' = a/\tau_u$ and $b' = b/\tau_v$. Second, if $Z = h^{a+b}$, then we clearly have a correctly distributed $K = \mathsf{Decap}(pk_1, sk_1, C^*) = (u^a)^x (v^b)^y$. Otherwise, if Z is random, so is K. □

5. VRF-Suitable IB-KEMs

In this section we describe our constructions of Verifiable Random functions from VRF-suitable IB-KEMs. In particular, in light of the results presented in Sect. 3, we focus on constructing VRF-suitable IB-KEM schemes.

We start by describing, in Sect. 5.1, a VRF from the Sakai–Kasahara IB-KEM [43]. Interestingly, the proposed VRF closely resembles the VRF proposed by Dodis and Yampolskiy [22].

Next, in Sect. 5.2, we present a new construction of VRF-suitable IB-KEM from the decisional ℓ-weak Bilinear Diffie–Hellman Inversion assumption (decisional ℓ-wBDHI*, following the acronym used in [9]), recalled in Sect. 2.1, that given $g, g^b, g^c, g^{b^2}, \ldots, g^{b^\ell}$, the quantity $e(g, g)^{b^{\ell+1}c}$ should remain indistinguishable from random to any polynomially bounded adversary. Interestingly, in order for our construction to work, the ℓ parameter does not need to be too large. This is because it only limits to 2^ℓ the size of the space of valid identities but it does not affect in any other way the number of adversarial queries allowed in the security proof (as in most known proofs using q-type assumptions). This means that it is enough to assume that the ℓ-wBDHI* assumption holds only for rather small values of ℓ (i.e. $\ell = 160$ or $\ell = 256$).

As a final note, we mention that, in principle, one could construct a VRF from Boneh–Franklin's IBE. Indeed, we prove in Appendix A.2, that the KEM version of the scheme is actually a VRF-suitable IB-KEM, under the decisional Bilinear Diffi–Hellman assumption. However, for the sake of building a VRF, this construction is of very limited interest as its proof holds in the random oracle model.

5.1. *Sakai–Kasahara VRF*

We briefly recall the KEM version of the Sakai–Kasahara IBE scheme (SKfor short) [43]. This scheme relies on the q-decisional Bilinear Diffie–Hellman Inversion assumption (DBDHI for short), which is defined in Sect. 2.1.4.

- $\mathsf{Setup}(1^k)$: The setup algorithm runs $\mathcal{G}(1^k)$ to obtain the description of the groups $\mathbb{G}, \mathbb{G}_T$ and of a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The description of $\mathbb{G}$ contains a generator $g \in \mathbb{G}$. Then the algorithm picks a random $s \xleftarrow{\$} \mathbb{Z}_p$ and sets $h = g^s$, $mpk = (g, h), msk = s$. For security reasons, the identity space $\mathcal{ID}$ is the subset of $\mathbb{Z}_p$ limited to the first q elements of $\mathbb{Z}_p$, where q is some polynomial in the security parameter.
- $\mathsf{KeyDer}(msk, ID)$: Let $ID \in \mathcal{ID}$. The key derivation algorithm constructs the secret key $sk_{ID} = g^{1/s+ID}$. In the unlikely case that $ID = -s$, we define sk_{ID} to be $1 \in \mathbb{G}$.
- $\mathsf{Encap}(mpk, ID)$: The encapsulation algorithm picks a random $t \xleftarrow{\$} \mathbb{Z}_p^\star$ and computes a random session key $K = e(g, g)^t$ and a corresponding ciphertext $C = (g^s g^{ID})^t$.
- $\mathsf{Decap}(mpk, ID, sk_{ID}, C)$: The decapsulation algorithm uses the secret key sk_{ID} to compute a session key K from a ciphertext C as follows: $K = e(C, sk_{ID})$.

First, notice that by assuming $aux_{ID} = \perp$ for all identities ID, the above description fits our syntax of VRF-suitable IB-KEMs. In the following theorem we prove that the Sakai–Kasahara IB-KEM scheme can be used to construct a VRF (i.e., that it actually provides unique decapsulation and pseudo-random decapsulation). Precisely, we first show that the scheme is selective-secure. Then its full-security follows by applying the result of Proposition 2.

Theorem 7. *Assuming that the q-DBDHI assumption holds in a bilinear group $\mathbb{G}$, then the Sakai–Kasahara IB-KEM* [43] *is a selective-secure VRF-suitable IB-KEM.*

Proof. We prove the theorem by showing that the IB-KEM scheme presented above has unique decapsulation and satisfies the selective pseudo-random decapsulation property.

First, we observe that unique decapsulation follows by construction. Indeed, if we fix a specific (mpk, msk) pair, then one can obtain only one key for each identity (i.e., no random choices are possible). More formally, consider the second condition of the unique decapsulation property, where one checks that both the keys sk_{ID} and sk'_{ID} decrypt correctly. Let $C = g^{(s+ID)t}$ and $K = e(g, g)^t$ be honestly generated ciphertext and session key. By the definition of the decapsulation algorithm, the equation $e(g^{(s+ID)t}, sk_{ID}) = e(g, g)^t$ holds if and only if $sk_{ID} = g^{1/s+ID}$. Hence, if this check holds for both sk_{ID} and sk'_{ID} (even for different t's), then it must be $sk_{ID} = sk'_{ID}$.

Now, let us focus on proving that SKsatisfies selective pseudo-random decapsulation under the DBDHI assumption. Let $\mathcal{ID} = \{ID_0, \ldots, ID_{q-1}\} \subseteq \mathbb{Z}_p$ be the sets of all possible identities (i.e. the first q elements of $\mathbb{Z}_p$), and let ID_0 be the zero-identity, i.e., $0 \in \mathbb{Z}_p$. Here we prove that SKsatisfies pseudo-random decapsulation w.r.t. the distribution $\mathcal{D}_{\mathcal{ID}}$ that always outputs $ID_0 = 0$. We stress that even such a restricted distribution is sufficient for instantiating our generic construction and building a VRF.

So, for the sake of contradiction, suppose there exists an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that has non-negligible advantage $\epsilon(k)$ into breaking the selective pseudo-random decapsulation of SK IB-KEM w.r.t. ID_0. Then we show how to build a simulator $\mathcal{B}$ which is able to break the DBDHI assumption with non-negligible advantage $\epsilon(k)$.

$\mathcal{B}$ receives in input a tuple $(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)}, Z) \in \mathbb{G}^{q+1} \times \mathbb{G}_T$ and must output 0 if it believes that $Z = e(g,g)^{1/x}$, or 1 otherwise. First, $\mathcal{B}$ runs $\mathcal{A}_1$ to obtain the challenge identity $ID_k \in \mathcal{ID}$. Let s be implicitly defined as $x - ID_k$. Using the binomial theorem $\mathcal{B}$ computes $(g, g^s, g^{(s^2)}, \ldots, g^{(s^q)})$. Then $\mathcal{B}$ defines the polynomial $f(z) = \prod_{i=0, i\neq k}^{q-1}(z + ID_i) = \sum_{i=0}^{q-1} z^i \beta_i$, and computes $g' = \prod_{i=0}^{q-1} g^{s^i \beta_i} = g^{f(s)}$ and $h' = \prod_{i=1}^{q-1} g^{s^i \beta_{i-1}} = g^{sf(s)} = (g')^s$. It picks a random $t \xleftarrow{\$} \mathbb{Z}_p$ and sets $C_0 = (g')^t$. We observe that C_0 is a valid ciphertext under identity ID_0 and randomness t/s.

At this point it is worth noting that with all but negligible probability the values g', h', C_0 perfectly simulate the real values. The only unlucky cases are when $g' = 1$ (i.e., $f(s) = 0 \bmod p$) or when $g' \neq 1$ and $h' = 1$ (i.e., $s = 0$). However, in both cases, it is easy to see that $\mathcal{B}$ can directly recover x and break the DBDHI assumption.

To complete the first part of the simulation, $\mathcal{B}$ computes a session key $\bar{K}$ as follows. Let $f'(z) = \frac{f(z)}{z+ID_k} - \frac{\gamma}{z+ID_k} = \sum_{i=0}^{q-2} z^i \gamma_i$, where $\gamma \neq 0$ is the remainder of the division of $f(z)$ by $z + ID$. First $\mathcal{B}$ computes

$$Z_0 = \left(\prod_{i=0}^{q-1}\prod_{j=0}^{q-2} e\left(g^{s^i}, g^{s^j}\right)^{\beta_i \gamma_j}\right)\left(\prod_{m=0}^{q-2} e\left(g, g^{s^m}\right)^{\gamma \gamma_m}\right) = e(g,g)^{\frac{f(s)^2 - \gamma^2}{x}}.$$

$\mathcal{B}$ sets $\bar{K} = (Z_0 \cdot Z^{\gamma^2})^t$.Then $\mathcal{B}$ gives $mpk = (g', h')$, C_0 and $\bar{K}$ to the adversary.

When $\mathcal{A}$ asks for the private key of an identity $ID_j \neq ID_k$ $\mathcal{B}$ computes the secret key in the following way. First it defines the polynomial $f_j(z) = \frac{f(z)}{z+ID_j} = \prod_{i=0, i\neq j,k}^{q-1}(z + ID_i) = \sum_{i=0}^{q-2} z^i \delta_i$. Then it computes $sk_{ID_j} = (g')^{1/s+ID_j} = g^{f(s)/s+ID_j} = g^{f_j(s)} = \prod_{i=0}^{q-2} g^{s^i \delta_i}$ and returns sk_{ID_j} to $\mathcal{A}$.

At the end of the experiment $\mathcal{A}$ is supposed to output its guess b'. $\mathcal{B}$ outputs the same b' as its guess for Z. Observe that if $Z = e(g,g)^{1/x}$, then $\mathcal{B}$ computed a session key of the correct form: $\bar{K} = e(g', g')^{\frac{t}{s+ID_k}}$. Otherwise, if Z is a random element of $\mathbb{G}_T$, then $\bar{K}$ will be random too.

In conclusion, $\mathcal{B}$ succeeds with the same probability as $\mathcal{A}$. However, due to the way the public parameters are generated, if $\mathcal{A}$ has a running time T, then $\mathcal{B}$'s running time is $O(T + q)$ where q is the size of the identity space. □

By applying the result of Proposition 2 to the previous theorem and to our transformation, we obtain the following Corollary.

Corollary 1. *Assuming that the q-DBDHI assumption holds in a bilinear group* $\mathbb{G}$, *then the VRF obtained from the Sakai–Kasahara VRF-suitable IB-KEM* [43] *is a fully-secure VRF for domains of polynomial size.*

We notice that the resulting VRF can only support an input space that is polynomially-sized (in the security parameter). This depends on two reasons. First, if we want a fully-

secure VRF, then the reduction of Proposition 2 has a security loss which is linear in the size of the input space. Second, even if we restrict only to selective-security, observe that the running time of the simulator in the security reduction of Theorem 7 is linear in the size of the input space.

We remark that all previously known constructions of VRFs [20,22,38,41] made the same restriction on the size of the input space.

Similarity with the Dodis–Yampolskiy VRF Here we show that the Dodis–Yampolskiy VRF [22] (that we briefly recall in Appendix B) closely resembles the construction obtained from our transformation. Indeed, Theorem 7 leads to the following VRF.

- $\mathsf{Gen}(1^k)$: The key generation algorithm runs $\mathcal{G}(1^k)$ to obtain the description of the groups $\mathbb{G}, \mathbb{G}_T$ and of a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The description of $\mathbb{G}$ contains a generator $g \in \mathbb{G}$. Then the algorithm picks random $s, t \xleftarrow{\$} \mathbb{Z}_p$ and sets $h = g^s$, $C_0 = h^t$, $vpk = (g, h, C_0)$, $vsk = s$.
- $\mathsf{Func}_{vsk}(x)$: Let $\mathsf{Func}_{vsk}(x)=(F_{vsk}(x), \pi_{vsk}(x))$. One sets $\mathsf{Func}_{vsk}(x) = e(C_0, sk_x) = e(g, g)^{(st)/(s+x)}$ as the VRF output and $\pi_{vsk}(x) = \mathsf{KeyDer}(x) = g^{1/(s+x)}$ as the proof of correctness.
- $\mathsf{V}(vpk, x, y, \pi_x)$: To verify whether y was computed correctly, one starts by running the Encap algorithm on input (vpk, x). Encap chooses $\omega \xleftarrow{\$} \mathbb{Z}_p$ and then computes $K \leftarrow e(g, g)^\omega$ and $C = (hg^x)^\omega$. Then one checks that $K = \mathsf{Decap}(mpk, x, \pi_x, C) = e((g^x \cdot h)^\omega, \pi_x)$ and $y = \mathsf{Decap}(mpk, x, \pi_x, C_0) = e(h^t, \pi_x)$.

By setting $t = s^{-1} \bmod p$ and $\omega = 1$, the construction above can be optimized to get exactly the Dodis–Yampolskiy VRF. It is worth noting, however, that our security analysis does not directly work with the optimized scheme.

5.2. *Our New Construction*

In this section we propose a new construction of a VRF-suitable IB-KEM from the (conjectured) computational intractability of the decisional weak ℓ-Bilinear Diffie–Hellman Inversion problem (see Sect. 2.1.5 for a formal description). The new scheme is inspired by Lysyanskaya's VRF [38] in that the validity of each new auxiliary information aux_{ID} (required to compute the session key) is verified by exploiting the DDH-CDH separation in bilinear groups. Our new scheme, however, is more efficient as it leads to a VRF directly (i.e., rather than having to construct a unique signature scheme first), and it does not require error correcting codes. The proposed scheme follows.

- $\mathsf{Setup}(1^k)$: The setup algorithm runs $\mathcal{G}(1^k)$ to obtain the description of the groups $\mathbb{G}, \mathbb{G}_T$ and of a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The description of $\mathbb{G}$ contains a generator $g \in \mathbb{G}$. Let $\{0, 1\}^\ell$ be the space of valid identities. Then the algorithm picks (at random) $a, \alpha_1, \beta_1, \ldots, \alpha_\ell, \beta_\ell \xleftarrow{\$} \mathbb{Z}_p$, sets $g_1 = g^a$, and for $i = 1, \ldots, \ell$ sets $g_{0i} = g^{\beta_i}$ and $g_{1i} = g^{\alpha_i}$. The public parameters are

$$mpk = \big(g, g_1, \{g_{ij}\}_{i=0,1;\, j=1..\ell}\big).$$

  The master secret key is $msk = (a, \{\alpha_i, \beta_i\}_{i=1,..,\ell})$.

– KeyDer(msk, ID): We assume $ID = ID_1 \cdots ID_\ell$ where each $ID_i \in \{0,1\}$. The key derivation algorithm constructs the secret key sk_{ID} and the auxiliary information aux_{ID} as follows. Let $h_0 = g$, for $i = 1$ to ℓ one computes

$$h_i = (h_{i-1})^{\alpha_i^{ID_i} \beta_i^{(1-ID_i)}}$$

and sets $aux_{ID} = (h_1, \ldots, h_\ell)$ and $sk_{ID} = h_\ell^a$.

– Encap(mpk, ID, aux_{ID}): Let $aux_{ID} = (h_1, \ldots, h_\ell)$ computed as above. The encapsulation algorithm picks a random $t \xleftarrow{\$} \mathbb{Z}_p^\star$ and computes a random session key $K = e(g_1, h_\ell)^t$ and a corresponding ciphertext $C = g^t$.

– Decap($mpk, ID, sk_{ID}, aux_{ID}, C$): The decapsulation algorithm uses the secret key sk_{ID} and the auxiliary information aux_{ID} to compute a session key K from a ciphertext C. This is done as follows. First, in order to guarantee the unique decapsulation property, a check on the validity of the auxiliary information has to be performed. This is done as follows. Let $h_0 = g$, for $i = 1, \ldots, \ell$

$$\textbf{if } ID_i = 1 \textbf{ check } e(g, h_i) \stackrel{?}{=} e(g_{1i}, h_{i-1})$$
$$\textbf{else} \quad \textbf{check } e(g, h_i) \stackrel{?}{=} e(g_{0i}, h_{i-1})$$

If any of the above checks fails output reject. Second, the key K is computed as $K = e(C, sk_{ID}) = e(g_1, h_\ell)^t$. Note that the validity of sk_{ID} can be verified by first encrypting some random message m with respect to the public key (g, g_1, h_ℓ) and then by checking if one can decrypt it correctly using sk_{ID}.

Security The following theorem states the security of our new scheme.

Theorem 8. *Suppose the decisional ℓ-wBDHI* assumption holds in $\mathbb{G}$, then the scheme given above is a selective-secure VRF-suitable IB-KEM scheme.*

Proof. Let $\mathcal{ID} = \{0,1\}^\ell$ the identity space. First note that the scheme fits the syntax of VRF-suitable IB-KEMs. We prove the theorem by showing that the scheme satisfies the unique decapsulation property and meets the selective pseudo-random decapsulation requirement.

Unique Decapsulation. We prove this by showing that for a given identity ID the corresponding h_ℓ is uniquely determined as

$$h_\ell = g^{\prod_{i=1}^{\ell} \alpha_i^{ID_i} \beta_i^{1-ID_i}}.$$

The proof is by induction on i. First note that it must be the case $h_1 = g^{\alpha_1^{ID_1} \beta_1^{1-ID_1}}$, as otherwise the check $e(g, h_1) \stackrel{?}{=} e(g_{ID_1 1}, h_0) = e(g^{\alpha_1^{ID_1} \beta_1^{1-ID_1}}, g)$ would fail. Now assume that the statement holds true for any index $j - 1 < \ell$, i.e. that $h_{j-1} = g^{\prod_{i=1}^{j-1} \alpha_i^{ID_i} \beta_i^{1-ID_i}}$. We prove that the same holds for j.

$$h_j = h_{j-1}^{\alpha_j^{ID_j} \beta_j^{1-ID_j}} = \left(g^{\prod_{i=1}^{j-1} \alpha_i^{ID_i} \beta_i^{1-ID_i}}\right)^{\alpha_j^{ID_j} \beta_j^{1-ID_j}} = g^{\prod_{i=1}^{j} \alpha_i^{ID_i} \beta_i^{1-ID_i}}.$$

SELECTIVE PSEUDO-RANDOM DECAPSULATION. We prove the selective pseudo-random decapsulation w.r.t. to any arbitrary distribution $\mathcal{D}_{\mathcal{ID}}$ over the identity space. Namely, our proof works for any ID_0 in the identity space $\mathcal{ID}$.

Assume that there is an adversary $\mathcal{A}$ that breaks the selective pseudo-random decapsulation of the proposed scheme with advantage ϵ, then we show how to build an adversary $\mathcal{B}$ that solves the decisional ℓ-wBDHI* problem with advantage ϵ and runs in time comparable to that needed by $\mathcal{A}$. $\mathcal{B}$ starts by receiving, from some challenging oracle, the values $(C = g^c, B_1 = g^b, B_2 = g^{b^2}, \ldots B_\ell = g^{b^\ell})$ and a value Z that can be either of the form $e(g,g)^{b^{\ell+1}c}$ or of the form $e(g,g)^z$, for random $z \in \mathbb{Z}_p^*$, depending on some random (and hidden) bit d that $\mathcal{B}$ is supposed to guess. First, note that in the proposed scheme the ciphertext C is independent of specific identities, thus $\mathcal{B}$ can produce it without having to commit to any ID_0. $\mathcal{B}$ gets the challenge identity $\overline{ID}$ as input from $\mathcal{A}$. Next, it sets $g_1 = B_1$, it chooses random $\alpha_i, \beta_i \xleftarrow{\$} \mathbb{Z}_p^*$, for $i = 1, \ldots, \ell$, and for $i = 1, \ldots, \ell$ it computes the following values:

$$g_{0i} = \begin{cases} B_1^{\beta_i} & \text{if } \overline{ID}_i = 0 \\ g^{\beta_i} & \text{if } \overline{ID}_i = 1 \end{cases} \qquad g_{1i} = \begin{cases} g^{\alpha_i} & \text{if } \overline{ID}_i = 0 \\ B_1^{\alpha_i} & \text{if } \overline{ID}_i = 1 \end{cases}$$

Note that the public parameters $mpk = (g, g_1, \{g_{ij}\}_{i=0,1;\, j=1..\ell})$ are distributed exactly as those produced by the setup algorithm. The master secret key is implicitly set to $msk = (b, \{\alpha_i b^{\overline{ID}_i}, \beta_i b^{1-\overline{ID}_i}\}_{i=1,..,\ell})$. Next, $\mathcal{B}$ computes C^* as follows $C^* \leftarrow C = g^c$. Thus, C^* is also correctly distributed. $\mathcal{B}$ constructs the challenge key $K_{\overline{ID}}$ by computing $Z^{\omega_{\overline{ID}}}$, where $\omega_{\overline{ID}} = \prod_{i=1}^{\ell} \alpha_i^{\overline{ID}_i} \beta_i^{1-\overline{ID}_i}$. The value $aux_{\overline{ID}}$ is computed as follows: h_ℓ is $B_\ell^{\omega_{\overline{ID}}}$ and h_i is $B_i^{\omega_{\overline{ID},i}}$ where $\omega_{\overline{ID},i} = \prod_{j=1}^{i} \alpha_j^{\overline{ID}_j} \beta_j^{1-\overline{ID}_j}$. Note that $\mathcal{B}$ is not able to explicitly compute $sk_{\overline{ID}} = B_{\ell+1}^{\omega_{\overline{ID}}}$. However, this is not a problem as $\mathcal{B}$ is not required to do so. $\mathcal{B}$ runs $\mathcal{A}$ on input $(mpk, C^*, ID_0, K_{\overline{ID}}, aux_{\overline{ID}})$, for some identity ID_0 chosen according to $\mathcal{D}_{\mathcal{ID}}$.

Now, we show how $\mathcal{B}$ can answer key derivation queries for identities $ID \neq \overline{ID}$. Since $ID \neq \overline{ID}$, there exists (at least) an index j such that $ID_j \neq \overline{ID}_j$. For such index we have that either $g_{0j} = g^{\beta_j}$ (if $ID_j = 0$) or $g_{1j} = g^{\alpha_j}$ (otherwise). This means that the h_ℓ corresponding to identity ID will contain the (unknown) b with exponent $\ell - 1$, at most. Let $n < \ell$ denote the number of positions i such that $ID_i = \overline{ID}_i$. $\mathcal{B}$ computes the h_i as follows.

$$h_1 = \begin{cases} g^{\alpha_1^{ID_1} \beta_1^{1-ID_1}} & \text{if } ID_1 \neq \overline{ID}_1 \\ B_1^{\alpha_1^{ID_1} \beta_1^{1-ID_1}} & \text{if } ID_1 = \overline{ID}_1 \end{cases}$$

$$h_2 = \begin{cases} h_1^{\alpha_2^{ID_2} \beta_2^{1-ID_2}} & \text{if } ID_2 \neq \overline{ID}_2 \\ B_1^{\alpha_2^{ID_2} \beta_2^{1-ID_2} \alpha_1^{ID_1} \beta_1^{1-ID_1}} & \text{if } ID_2 = \overline{ID}_2 \wedge ID_1 \neq \overline{ID}_1 \\ B_2^{\alpha_2^{ID_2} \beta_2^{1-ID_2} \alpha_1^{ID_1} \beta_1^{1-ID_1}} & \text{if } ID_2 = \overline{ID}_2 \wedge ID_1 = \overline{ID}_1 \end{cases} \quad \cdots$$

Finally, letting $\omega_{\mathcal{ID}} = \prod_{i=1}^{\ell} \alpha_i^{ID_i} \beta_i^{1-ID_i}$, h_ℓ is computed as $B_n^{\omega_{ID}}$.

The secret key sk_{ID} is computed as $B_{n+1}^{\omega_{ID}}$. Recall that, since $n < \ell$, $\mathcal{B}$ can do this operation using the values received by the challenger. It is easy to check that both the $aux_{ID} = (h_1, \ldots, h_\ell)$ and sk_{ID} are distributed as in the real key derivation algorithm.

At the end of the game, $\mathcal{A}$ returns a bit d' ($d' = 0$ means real, $d' = 1$ means random), and $\mathcal{B}$ outputs the same d'. This completes the description of the simulator.

Now notice that if $Z = e(g, g)^{b^{\ell+1}c}$, $K_{\overline{ID}}$ is a valid key for the identity $\overline{ID}$. This is because $K_{\overline{ID}} = e(g_1, h_{\overline{ID}})^c$, where $h_{\overline{ID}}$ is the h_ℓ corresponding to identity $\overline{ID}$. Thus, $h_{\overline{ID}} = g^{b^\ell \omega_{\overline{ID}}}$.

$$K_{\overline{ID}} = e(g_1, h_{\overline{ID}})^c = e\big(g^b, g^{b^\ell \omega_{\overline{ID}}}\big)^c = Z^{\omega_{\overline{ID}}}.$$

If, on the other hand, Z is a random value so is $K_{\overline{ID}}$. Thus, by standard calculations one gets that, if $\mathcal{A}$ has advantage ϵ in breaking the (selective) pseudo-random decapsulation property of the scheme, $\mathcal{B}$ breaks the decisional ℓ-wBDHI* with advantage ϵ. □

Remark 4. It is interesting to note that the above theorem shows that our scheme satisfies the selective-notion without any restriction on the size of the input space. This does not hold for the Dodis–Yampolskiy VRF because in the security proof (even for selective security) the running time of the simulator is linear in the size of the input space.

The Resulting VRF We briefly show the VRF construction that results from applying our transformation to the VRF-suitable IB-KEM scheme described above.

- $\mathsf{Gen}(1^k)$: The key generation algorithm runs $\mathcal{G}(1^k)$ to obtain the description of the groups $\mathbb{G}, \mathbb{G}_T$ and of a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The description of $\mathbb{G}$ contains a generator $g \in \mathbb{G}$. Let $\{0,1\}^\ell$ be the input space. The algorithm picks $t, a, \alpha_1, \beta_1, \ldots, \alpha_\ell, \beta_\ell \xleftarrow{\$} \mathbb{Z}_p$, uniformly at random and it sets $g_1 = g^a$, $C = g^t$. Next, for $i = 1, \ldots, \ell$ it computes $g_{0i} = g^{\beta_i}$ and $g_{1i} = g^{\alpha_i}$. The public key is

  $$vpk = \big(g, g_1, C, \{g_{ij}\}_{i=0,1;\, j=1..\ell}\big)$$

  whereas the secret key is $vsk = (a, \{\alpha_i, \beta_i\}_{i=1,\ldots,\ell})$.
- $\mathsf{Func}(vsk, x)$: Let (y, π_x) be its output, and assume $x = x_1 \cdots x_\ell$ where each $x_i \in \{0,1\}$. The proof is constructed as follows. Let $h_0 = g$, for $i = 1$ to ℓ compute

  $$h_i = (h_{i-1})^{\alpha_i^{x_i} \beta_i^{(1-x_i)}}.$$

  Set $\pi_x = (h_1, \ldots, h_\ell, h_\ell^a)$.

  Instead, the VRF output is computed as $y = e(C, h_\ell^a)$.
- $\mathsf{V}(vpk, x, y, \pi_x)$: To verify whether y was computed correctly, proceed as follows: Let $\pi_x = (h_1, \ldots, h_\ell, h_\ell^a)$. Let $h_0 = g$, for $i = 1, \ldots, \ell$.

  $$\textbf{if } x_i = 1 \textbf{ check } e(g, h_i) \stackrel{?}{=} e(g_{1i}, h_{i-1})$$

  $$\textbf{else} \quad \textbf{check } e(g, h_i) \stackrel{?}{=} e(g_{0i}, h_{i-1})$$

  Check that $y = e(C, h_\ell^a)$. If any of the above checks fails output reject.

Efficiency and Security Considerations In terms of efficiency, we note that the resulting VRF scheme has public keys and proofs whose sizes are linear in the bit-length ℓ of the VRF's input. Likewise, the time needed to compute and verify a proof is also linear in ℓ. Clearly, these parameters are worse than those achieved by the Dodis–Yampolskiy VRF scheme, which enjoys constant-size proofs and public keys.

In terms of security, our new construction enjoys a tight reduction to the ℓ-wBDHI* assumption and is proven to be selective-secure for *large input spaces*. This is not the case for the Dodis–Yampolskiy VRF scheme, as its security reduction is based on the ℓ-BDHI assumption, where ℓ is linear in the size of the input space. Finally, as we point out in the following section, our new construction can also be proven fully-secure for large input spaces.

5.3. *A Scheme Secure for Large Identity Spaces*

Given the result of Theorem 8, if one wants to obtain a fully-secure VRF, one should apply Proposition 2 at the cost of losing a factor 2^ℓ in the final security (where 2^ℓ represents the size of the identity space). This means that the previous scheme is fully-secure only when the identity space is small (i.e., 2^ℓ is polynomial in the security parameter), or when the security parameter used to instantiate the bilinear groups is made large enough to have a significant reduction. However, the last solution unfortunately leads to parameters that are quite inefficient in practice.

In this section, we show that our scheme described in Sect. 5.2 can be proven secure without any exponential loss, even when the identity space is exponentially large, meaning that we can efficiently support large identity spaces. In the following sections, we show how to achieve this result using two different techniques: one is based on the notion of admissible hash functions [5], and the other uses the artificial abort technique introduced by Waters [46]. More precisely, in the first case we need to make a small modification in our scheme: we assume that each identity is a binary string of w bits, and that the scheme (originally working with ℓ-bits long identities) first hashes the identities using an admissible hash function $H : \{0,1\}^w \to \{0,1\}^\ell$. On the other hand, in order to use the artificial abort technique, we do not have to make any changes in our scheme of Sect. 5.2, but the security will hold under a slightly different assumption.

Full Security via Admissible Hash Functions The notion of admissible hash functions was first introduced by Boneh and Boyen in [5] as a tool for proving the full security of their identity-based encryption scheme. These functions have been shown to be useful in order to secretly partition the identity space in two subsets, the *blue* set and the *red* set, so that there is a noticeable probability that all the adversary's secret key queries fall in the blue set and the challenge identity is in the red set. This fact is particularly useful in those reductions where the simulator can be programmed so that it is able to answer secret key queries for blue identities, while it can generate a challenge ciphertext for any red identities. Boneh and Boyen showed a construction of admissible hash functions exist based on collision-resistance and error correcting codes.

In our work, we use admissible hash functions in a similar way. In particular, we choose to follow the definition by Cash *et al.* [14,15] as it looks easier to use.

Let $k \in \mathbb{N}$ be the security parameter, w and ℓ be two values polynomial in k. Let $\mathcal{H} = \{H : \{0,1\}^w \to \{0,1\}^\ell\}$ be a family of functions. For $H \in \mathcal{H}$, $V \in \{0,1,\perp\}^\ell$ and

any $x \in \{0,1\}^w$ we define:

$$F_{V,H}(x) = \begin{cases} \mathtt{B} & \text{if } \exists i \in \{1,\ldots,\ell\} : H(x)_i = V_i \\ \mathtt{R} & \text{if } \forall i \in \{1,\ldots,\ell\} : H(x)_i \neq V_i \end{cases}$$

For $m \in \{0,\ldots,\ell\}$, we denote by $\mathcal{V}^{(\ell,m)}$ the uniform distribution over $\{0,1,\perp\}^\ell$ such that exactly m components are in $\{0,1\}$.

Definition 6. $\mathcal{H} = \{H : \{0,1\}^w \to \{0,1\}^\ell\}$ is a family of *Δ-admissible hash functions* if, for every polynomial $Q = Q(k)$, there exists an efficiently computable function $m = m(k)$ and efficiently recognizable sets $bad_H \subseteq (\{0,1\}^w)^*$ such that the following properties hold:

1. For every PPT algorithm $\mathcal{A}$ that, on input $H \in \mathcal{H}$, outputs $\boldsymbol{x} \in (\{0,1\}^w)^{Q+1}$, the following advantage is a negligible function in k:

$$\mathbf{Adv}^{\mathrm{adm}}_{\mathcal{H},\mathcal{A}}(k) = \Pr\big[\boldsymbol{x} \in bad_H : H \leftarrow \mathcal{H}, \boldsymbol{x} \leftarrow \mathcal{A}(H)\big].$$

2. For every $H \in \mathcal{H}$, $V \leftarrow \mathcal{V}^{(\ell,m)}$, and every vector $x \in (\{0,1\}^w)^{Q+1} \setminus bad_H$ we have

$$\Pr\big[F_{V,H}(x_0) = \mathtt{R} \wedge F_{V,H}(x_1) = \mathtt{B} \wedge F_{V,H}(x_2) = \mathtt{B} \wedge \cdots \wedge F_{V,H}(x_Q) = \mathtt{B}\big] \geq \Delta(k, Q).$$

$\mathcal{H}$ is said *admissible* if it is Δ-admissible for some Δ such that $\Delta(k, Q)$ is significant for every $Q = Q(k)$.

Once we have defined the notion of admissible hash functions, we consider the scheme given in Sect. 5.2 modified as follows. We let the identities be strings $I \in \{0,1\}^w$, and we use our scheme of Sect. 5.2 by hashing every identity I to $ID = H(I) \in \{0,1\}^\ell$ using a function H taken from an admissible family $\mathcal{H} = \{H : \{0,1\}^w \to \{0,1\}^\ell\}$. To concretely instantiate this scheme, one can use the construction of admissible hash functions proposed by Boneh and Boyen in [5], whose security relies on collision-resistant hash functions. We defer the interested reader to [5, Sect. 5.3] and [15, Sect. 5.4.4] for a more precise description of the construction and the possible choices of parameters.

We now prove the following theorem.

Theorem 9. *Suppose the decisional ℓ-wBDHI* assumption holds in $\mathbb{G}$ and $\mathcal{H}$ is a family of admissible hash functions, then the scheme of Sect. 5.2 with the above modifications is a secure VRF-suitable IB-KEM.*

Proof. First of all, observe that the unique decapsulation property holds for the same reasons given in Theorem 8. Therefore, it only remains to prove the pseudo-random decapsulation property. As in the proof of Theorem 8, we prove pseudo-random decapsulation w.r.t. to any arbitrary distribution $\mathcal{D}_{\mathcal{ID}}$ over the identity space (i.e., our proof works for any $ID_0 \in \mathcal{ID}$).

We prove the theorem by describing a series of games. Let $\overrightarrow{ID} \in (\{0,1\}^w)^{Q+1}$ be the set of identities queried by the adversary such that the first element of this vector is the challenge identity ID^*. For any i, we denote by G_i the output of Game i.

Game 0 is the real pseudo-random decapsulation experiment $\mathbf{Exp}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A}}(k)$. By definition we know that:

$$\mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{IBKEM},\mathcal{A}}(k) = \left|\Pr[G_0 = 1] - \frac{1}{2}\right|.$$

Game 1 is the same experiment as Game 0 except that in Game 1 the challenger aborts and outputs a random bit if $\overrightarrow{ID} \in bad_H$. By the first condition of admissible hash functions, it is easy to show that any adversary distinguishing Game 0 and Game 1 can be reduced to an adversary $\mathcal{C}$ against the admissibility of $\mathcal{H}$. Thus we have:

$$\left|\Pr[G_1 = 1] - \Pr[G_0 = 1]\right| \leq \mathbf{Adv}^{\text{adm}}_{\mathcal{H},\mathcal{C}}(k).$$

Game 2 proceeds as Game 1 except that at the end of the experiment, the challenger generates an event $good_2$ with probability Δ, and it aborts if $good_2$ does *not* occur. Thus we have:

$$\left|\Pr[G_2 = 1] - \frac{1}{2}\right| = \Pr[good_2]\left|\Pr[G_1 = 1] - \frac{1}{2}\right|.$$

Game 3 proceeds as Game 2 except for the following change at the end of the experiment. Instead of generating the event $good_2$, in Game 3 the challenger chooses a vector $V \leftarrow \mathcal{V}^{(\ell,m)}$ and computes $F_{V,H}(ID)$ for every identity ID queried by the adversary. Let E be the event

$$\text{“}F_{V,H}\left(ID^*\right) = \mathtt{R} \wedge F_{V,H}(ID_1) = \mathtt{B} \wedge F_{V,H}(ID_2) = \mathtt{B} \wedge \cdots \wedge F_{V,H}(ID_Q) = \mathtt{B}\text{”}.$$

Since $\overrightarrow{ID} \notin bad_H$, by the second condition of the admissibility of H, we have $\Pr[E] \geq \Delta$.

Next, the challenger samples $\lceil kS^2/\Delta^2 \rceil$ vectors $\tilde{V}$, where $S = poly(k)$ is an arbitrary polynomial, and, for each of these samples, it evaluates the function $F_{\tilde{V},H}$ on the given set of identities $\overrightarrow{ID}$. In this way, the challenger computes an approximation $\tilde{p}_E$ of $p_E = \Pr[E|\overrightarrow{ID}]$.

Finally, if E does not occur, then the challenger in Game 3 aborts. But even if E occurs, then it aborts with probability $1 - \Delta/\tilde{p}_E$ (recall that in the case of abort, the experiment outputs a random bit). Let $good_3$ be the event that Game 3 does not abort. Then we have:

$$\Pr[good_3] = \Delta \cdot \frac{p_E}{\tilde{p}_E}.$$

To analyze the difference between Game 2 and Game 3, one may think about directly replacing the event $good_2$ with the event E. However, as noticed by Cash *et al.* in their proof [14], this is not possible as the event E may not be independent of the adversary's view. More precisely, E is conditioned on the set of identities $\overrightarrow{ID}$ asked

by $\mathcal{A}$. To solve the issue, the game is modified by adding an artificial abort step whose goal is to make the overall abort probability sufficiently independent of $\mathcal{A}$'s view. So, to complete this analysis, first notice that by Hoeffding's inequality, $\lceil kS^2/\Delta^2 \rceil$ samples are sufficient to lower bound $\tilde{p}_E \geq \Delta$ in such a way that

$$\Pr\left[|p_E - \tilde{p}_E| \geq \frac{\Delta}{S}\right] \leq \frac{1}{2^k}.$$

Therefore, we obtain that the difference

$$\left|\Pr[good_3] - \Pr[good_2]\right| = \Delta \cdot \left|\frac{\tilde{p}_E - p_E}{\tilde{p}_E}\right| \leq \frac{\Delta}{S}$$

holds with probability $1 - 1/2^k$, and thus we have

$$\left|\Pr[G_3] - \Pr[G_2]\right| \leq \frac{\Delta}{S} + \frac{1}{2^k}.$$

REDUCING GAME 3 TO ℓ-*wBDHI**. The final step of the proof is to show that $|\Pr[G_3 = 1] - 1/2| \leq \mathbf{Adv}_{\mathcal{B}}^{\ell\text{-wBDHI}^*}(k)$. For the sake of contradiction, assume there exists an adversary $\mathcal{A}$ who wins in Game 3 with advantage ϵ, then we show how to build an adversary $\mathcal{B}$ that solves the decisional ℓ-wBDHI* problem with the same advantage and runs in time comparable to that needed by $\mathcal{A}$. $\mathcal{B}$ receives $(C = g^c, B_1 = g^b, B_2 = g^{b^2}, \ldots B_\ell = g^{b^\ell})$ and a value Z that can be either of the form $e(g,g)^{b^{\ell+1}c}$ or of the form $e(g,g)^z$, for random $z \in \mathbb{Z}_p^*$, depending on some random (and hidden) bit d that $\mathcal{B}$ is supposed to guess. $\mathcal{B}$ sets $g_1 = B_1$, and it chooses $V \leftarrow \mathcal{V}^{(\ell,m)}$ and random exponents $\alpha_i, \beta_i \xleftarrow{\$} \mathbb{Z}_p^*$, for $i = 1, \ldots, \ell$, and computes for $i = 1, \ldots, \ell$

$$g_{0i} = \begin{cases} g^{\beta_i} & \text{if } V_i = 0 \\ B_1^{\beta_i} & \text{if } V_i = 1 \text{ or } V_i = \perp \end{cases} \qquad g_{1i} = \begin{cases} B_1^{\alpha_i} & \text{if } V_i = 0 \text{ or } V_i = \perp \\ g^{\alpha_i} & \text{if } V_i = 1 \end{cases}$$

Note that the public parameters $mpk = (g, g_1, \{g_{ij}\}_{i=0,1;j=1..\ell})$ are distributed exactly as those produced by the setup algorithm. Next, $\mathcal{B}$ sets $C^* \leftarrow C = g^c$. Thus, C^* is also correctly distributed. Now $\mathcal{B}$ runs $\mathcal{A}$ on input (mpk, C^*, ID_0), for an identity ID_0 chosen according to $\mathcal{D}_{\mathcal{ID}}$. In particular, ID_0 can be any identity in $\mathcal{ID}$.

The simulation is almost the same as that given in the proof of Theorem 8. The main difference is that here the simulator might abort. Let us show that a key derivation query can be answered as long as $F_{V,H}(ID) = \mathtt{B}$, whereas the simulator can generate a challenge for any identity ID^* such that $F_{V,H}(ID^*) = \mathtt{R}$.

For key derivation queries, note that when $F_{V,H}(ID) = \mathtt{B}$ there always exists an index i such that $H(ID)_i = V_i$. For such index we have that either $g_{0i} = g^{\beta_i}$ (if $H(ID)_i = 0$) or $g_{1i} = g^{\alpha_i}$ (otherwise). This means that the value h_ℓ corresponding to the identity ID will contain the (unknown) b with exponent $\ell - 1$, at most. Thus, it is easy to see that the secret key and the auxiliary information can be efficiently computed.

On the other hand, observe that when $F_{V,H}(ID^*) = \mathtt{R}$, then ID^* disagrees with V in all positions and thus h_ℓ contains the unknown b with exponent exactly ℓ. So, the simulator can plug the value Z into the challenge session key.

Finally, the simulator runs the artificial abort step like the challenger in Game 3, and, if no abort condition occurs, then it outputs the same bit returned by $\mathcal{A}$. It is easy to see that the view obtained by the adversary in the simulation provided by $\mathcal{B}$ is distributed exactly as the view obtained in Game 3, and thus we have that

$$\mathbf{Adv}_{\mathcal{B}}^{\ell\text{-wBDHI}^*}(k) \geq \left|\Pr[G_3 = 1] - 1/2\right|.$$

If we put together all the bounds showed before, then we have shown that for every PPT adversary $\mathcal{A}$ that makes at most a polynomial number $Q = Q(k)$ of key derivation queries in the IB-KEM-RDECAP experiment against the scheme $\mathcal{IBKEM}$, and for every polynomial $S = S(k)$, there exists an algorithm $\mathcal{B}$ against the ℓ-wBDHI* assumption and an algorithm $\mathcal{C}$ against the admissibility of $\mathcal{H}$ such that:

$$\mathbf{Adv}_{\mathcal{IBKEM},\mathcal{A}}^{\text{IB-KEM-RDECAP}}(k) \leq \mathbf{Adv}_{\mathcal{H},\mathcal{C}}^{\text{adm}}(k) + \frac{\mathbf{Adv}_{\mathcal{B}}^{\ell\text{-wBDHI}^*}(k)}{\Delta} + \frac{1}{S} + \frac{1}{2^k}. \qquad \square$$

Full Security via Artificial Abort In this section we show an alternative proof of security for the scheme of Sect. 5.2 that supports large identity spaces. We stress that in this case we do not make any changes to the original scheme (which is exactly the same as that shown in Sect. 5.2), but the security is proven under a slightly different assumption: the n-Decisional Diffie–Hellman Exponent assumption, originally introduced by Boneh et al. [10] and recalled below.

The n-Decisional Diffie–Hellman Exponent assumption (n-DDHE for short) is defined in bilinear groups $\mathbb{G}, \mathbb{G}_T$ of prime order p where there is a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $g, h \in \mathbb{G}$ be two generators and $b \in \mathbb{Z}_p^*$ be chosen at random.

We define the advantage $\mathbf{Adv}_{\mathcal{A}}^{n\text{DDHE}}(k)$ of a PPT algorithm $\mathcal{A}$ in solving n-DDHE in $\mathbb{G}$ as

$$\left|\Pr\left[c' = c \;\middle|\; \begin{array}{l} g, h \xleftarrow{\$} \mathbb{G}; b \xleftarrow{\$} \mathbb{Z}_p^*; \\ c \xleftarrow{\$} \{0,1\}; Z_0 \leftarrow e(g,h)^{b^n}; Z_1 \xleftarrow{\$} \mathbb{G}_T \\ c' \leftarrow \mathcal{A}(g, h, g^b, g^{b^2}, \ldots, g^{b^{n-1}}, g^{b^{n+1}}, \ldots, g^{b^{2n}}, Z_c) \end{array}\right] - \frac{1}{2}\right|.$$

Definition 7 (n-DDHE [10]). We say that the n-DDHE assumption holds in bilinear groups $\mathbb{G}, \mathbb{G}_T$ if, for any n polynomial in k, any PPT algorithm $\mathcal{A}$ has advantage $\mathbf{Adv}_{\mathcal{A}}^{n\text{DDHE}}(k)$ at most negligible in k.

We can now state the following theorem to prove the full security of our scheme. Its proof follows very closely the proof of security of the Hohenberger–Waters VRF [32], and thus we do not give it here. However, for completeness, we provide it in Appendix C.

Theorem 10. *Suppose the n-DDHE assumption holds in $\mathbb{G}$, then the scheme of Sect. 5.2 is a secure VRF-suitable IB-KEM.*

6. Conclusions

In this paper we introduced a new methodology to construct verifiable random functions from a class of identity-based key encapsulation schemes that we call VRF-suitable. We showed the applicability of our methods by providing two concrete realizations of the new primitive. The first one leads to a VRF that is very similar to the Dodis–Yampolskiy construction, while the second one leads to a new VRF. Moreover, the VRF resulting from our second construction enjoys the desired property of being fully-secure while efficiently supporting exponentially-large (in the security parameter) input spaces.

We observe that all known VRFs supporting large input spaces (ours as well as the schemes in [11,33]) require proofs containing roughly ℓ group elements where ℓ is length of the identity string, and their proofs hold *all* under q-type assumptions. We therefore believe that a natural and very intriguing question left open by this research is to find more efficient instantiations of VRFs for large input spaces, possibly ones provably secure under constant-size assumptions.

Acknowledgements

We thank Gregory Neven for collaborating with us at an early stage of this research. We also thank Eike Kiltz and Jonathan Katz for helpful discussions. The work of the second author was partially done while visiting the Computer Science Department at École Normale Supérieure. The third author did this work while he was student at University of Catania, and later while working at ENS. This work was supported in part by the European Commission through the IST Program under Contract ICT-2007-216646 ECRYPT II and in part by the French National Research Agency through the PACE project.

Appendix A. Examples of Schemes with Pseudo-random Decapsulation

A.1. *Pseudo-random Decapsulation of the Waters IB-KEM*

In this section we prove that the IBE scheme given by Waters in [46] satisfies the pseudo-random decapsulation property claimed in Sect. 2. More precisely, we focus on the KEM version of the scheme ($\mathcal{WKEM}$) and we prove that it has pseudo-random decapsulation assuming the Decisional Bilinear Diffie–Hellman assumption (DBDH for short) and that the scheme is secure against adaptively-chosen plaintext attacks (IB-KEM-CPA).

First we describe the Waters IB-KEM scheme ($\mathcal{WKEM}$) [46]. Let $\mathbb{G}, \mathbb{G}_T$ be two groups of the same order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We assume that identities are strings of length n.

– Setup: The setup algorithm picks a random generator $g \xleftarrow{\$} \mathbb{G}$ and a random $\alpha \xleftarrow{\$} \mathbb{Z}_p$. It sets $g_1 = g^\alpha$. Then it picks random elements $g_2, u', u_1, \ldots, u_n \xleftarrow{\$} \mathbb{G}$. The master public key is $mpk = (g, g_1, g_2, u', \{u_i\}_{i=1}^n)$, while the master secret key is $msk = g_2^\alpha$.

– $\mathsf{KeyDer}_{msk}(ID)$: The key derivation algorithm takes as input an identity $ID \in \{0,1\}^n$ and computes its related secret key d_{ID}. It picks a random $r \xleftarrow{\$} \mathbb{Z}_p$ and sets

$$d_{ID} = (d_1, d_2) = \left(g_2^{\alpha} \left(u' \prod_{i=1}^{n} u_i^{ID_i} \right)^r, g^r \right).$$

– $\mathsf{Encap}_{mpk}(ID)$: The encapsulation algorithm takes as input an identity ID and produces a session key K together with a ciphertext C. It picks a random $t \xleftarrow{\$} \mathbb{Z}_p$. Then it sets $C = (C_1, C_2) = (g^t, (u' \prod_{i=1}^{n} u_i^{ID_i})^t)$ and $K = e(g_1, g_2)^t$.
– $\mathsf{Decap}_{msk}(C, d_{ID})$: The decapsulation algorithm takes as input a ciphertext C and a secret key $d_{ID} = \mathsf{KeyDer}(msk, ID)$, and it computes the session key $K = \frac{e(d_1, C_1)}{e(d_2, C_2)}$.

Now we prove that such scheme has pseudo-random decapsulation w.r.t. the distribution $\mathcal{D}_{\mathcal{ID}}$ that always outputs $ID_0 = 0^n$.

Theorem 11. *If the DBDH assumption holds in $\mathbb{G}, \mathbb{G}_T$, then the scheme $\mathcal{WKEM}$ satisfies pseudo-random decapsulation.*

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary that has advantage $\mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{A}, \mathcal{WKEM}}(k) = \epsilon(k)$ against the pseudo-random decapsulation of $\mathcal{WKEM}$. We show how to construct a simulator $\mathcal{B}$ that with advantage at least $\epsilon(k)/2$ breaks either the standard IB-KEM-CPA security of $\mathcal{WKEM}$, or the DBDH assumption (recalled in Sect. 2.1.3).

Let $ID_0 = 0^n$ be the zero identity and let $\overline{ID}$ be the challenge identity chosen by the adversary $\mathcal{A}_1$. We distinguish two cases:

1. $\overline{ID} = ID_0$;
2. $\overline{ID} \neq ID_0$.

$\mathcal{A}_1$ will output a challenge identity of either type 1 or type 2. We describe two different simulations: in the first case we show how to break the security of the scheme $\mathcal{WKEM}$, whereas in the second case we show how to break the DBDH assumption.

So, at the beginning of the simulation $\mathcal{B}$ flips a binary coin $\beta \xleftarrow{\$} \{0,1\}$ and runs Simulation β as described below. Basically, if $\beta = 0$ $\mathcal{B}$ guesses that $\mathcal{A}$ will output a challenge identity of type 1. Otherwise, if $\beta = 1$ it guesses that $\mathcal{A}$ will output $\overline{ID} \neq ID_0$. It is easy to verify that the two simulations are perfectly indistinguishable from $\mathcal{A}$'s perspective.

SIMULATION 0. If $\beta = 0$ $\mathcal{B}$ acts as an adversary for the IB-KEM-CPA security of $\mathcal{WKEM}$. $\mathcal{B}$ receives mpk from its challenger and returns ID_0 as challenge identity. Then it gets back a ciphertext C_0 and a session key K^* and it runs $\mathcal{A}_1$ on input (mpk, C_0). $\mathcal{A}_1$ can issue key derivation queries until it outputs a challenge identity $\overline{ID}$. $\mathcal{B}$ answers such queries by using its key derivation oracle. If $\mathcal{A}$ asks for the private key of identity ID_0 or it outputs $\overline{ID} \neq ID_0$ then the simulator fails and outputs a random bit. Otherwise if $\mathcal{B}$ does not fail (and thus $\overline{ID} = ID_0$), it runs $b \leftarrow \mathcal{A}_2(K^*)$ and outputs b. Note that in this case breaking the pseudo-random decapsulation is equivalent to breaking the IB-KEM-CPA security.

Let fail denote the event that the simulator fails. Then the advantage of $\mathcal{B}$ against the security of $\mathcal{WKEM}$ is

$$\mathbf{Adv}^{\text{IB-KEM-CPA}}_{\mathcal{B},\mathcal{WKEM}}(k) = \mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{A},\mathcal{WKEM}}(k)\Pr[\overline{\text{fail}}].$$

SIMULATION 1. In this case $\mathcal{B}$ acts as an adversary for the Decisional Bilinear Diffie–Hellman assumption (DBDH). $\mathcal{B}$ receives in input a DBDH tuple (g, g^a, g^b, g^c, Z) from its challenger.

First, $\mathcal{B}$ constructs the public key for $\mathcal{WKEM}$ as follows. It chooses random values $\alpha, w, w_1, \ldots, w_n \xleftarrow{\$} \mathbb{Z}_p$ and $g_2 \in \mathbb{G}$. Then it sets $u' = g^w, u_i = (g^b)^{w_i}, g_1 = g^\alpha$. The public key is $mpk = (g, g_1, g_2, u', u_1, \ldots, u_n)$. The master secret key is $msk = \alpha$. $\mathcal{B}$ also computes the ciphertext $C_0 = (g^c, g^{cw})$. Note that C_0 is a correctly distributed ciphertext for the identity ID_0 and randomness c.

$\mathcal{B}$ runs $\mathcal{A}_1$ on input (mpk, C_0) and it answers key derivation queries using α until the adversary outputs the challenge identity $\overline{ID}$. If $\overline{ID} = ID_0$ then $\mathcal{B}$ aborts and outputs a random bit $b' \in \{0, 1\}$. Otherwise let $x = \sum_{i=1}^{n} w_i \overline{ID}_i$ and let $d_{\overline{ID}} = (d_1, d_2)$ be the secret key for the identity $\overline{ID}$. More precisely we have $d_1 = g_2^\alpha g^{wa} \prod_{i=1}^{n} g^{abw_i\overline{ID}_i}$ and $d_2 = g^a$. Note that $d_{\overline{ID}}$ is a correctly distributed secret key for the identity $\overline{ID}$ with randomness $r = a$, even if we are not able to compute d_1. Then $\mathcal{B}$ computes the session key $\bar{K} = e(g_2^\alpha g^{aw}, g^c)Z^x / e(g^a, g^{cw})$ and gives it to $\mathcal{A}_2$. In the end of the game $\mathcal{A}_2$ will output a bit b as its guess for $\bar{K}$. Then $\mathcal{B}$ will output the same b as its guess for Z.

We observe that if $Z = e(g, g)^{abc}$, $\bar{K}$ has the right form

$$\bar{K} = \frac{e(g_2^\alpha g^{aw}, g^c)e(g^{abx}, g^c)}{e(d_2, C_{0,2})} = \frac{e(g_2^\alpha g^{aw} g^{abx}, g^c)}{e(d_2, C_{0,2})} = \frac{e(d_1, C_{0,1})}{e(d_2, C_{0,2})}.$$

Otherwise, if Z is random, then so is $\bar{K}$. Let fail denote the event that the simulator fails. If $\mathcal{B}$ does not fail the simulation is perfect, thus its advantage against DBDH is

$$\mathbf{Adv}^{\text{DBDH}}_{\mathcal{B}}(k) = \mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{A},\mathcal{WKEM}}(k)\Pr[\overline{\text{fail}}].$$

In conclusion, since the two simulations are indistinguishable, the probability that $\mathcal{B}$ guesses correctly the type of the challenge identity output by $\mathcal{A}$ is $1/2$. Therefore, $\mathcal{B}$ breaks either the security of $\mathcal{WKEM}$ or the DBDH problem with advantage at least $\epsilon(k)/2$. □

A.2. *Pseudo-random Decapsulation of the Boneh–Franklin IB-KEM*

Let $\mathcal{BFKEM}$ refer to the KEM scheme derived from the BasicIdent IBE scheme by Boneh and Franklin [7]. Let $\mathbb{G}$ be a bilinear group of order p with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $H_1 : \{0, 1\}^* \to \mathbb{G}$ be a hash function. The description of $\mathcal{BFKEM}$ is as follows.

– Setup: The setup algorithm picks a random generator $P \in \mathbb{G}$ and a random $s \xleftarrow{\$} \mathbb{Z}_p$. It sets $P_{pub} = P^s$. The master public key is $mpk = (P, P_{pub}, H_1)$. The master secret is $msk = s$.

- $\mathsf{KeyDer}_{msk}(ID)$: The key derivation algorithm takes as input an identity $ID \in \{0,1\}^*$ and produces its related secret key d_{ID}. First it computes $Q_{ID} = H_1(ID)$ and then it sets $d_{ID} = Q_{ID}^s$.
- $\mathsf{Encap}_{mpk}(ID)$: The encapsulation algorithm takes as input an identity $ID \in \{0,1\}^*$ and returns a session key K together with a ciphertext C. The algorithm picks a random $r \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $Q_{ID} = H_1(ID)$. It sets $K = e(Q_{ID}, P_{pub})^r, C = P^r$.
- $\mathsf{Decap}_{msk}(C, d_{ID})$: The decapsulation algorithm takes as input a ciphertext C and a secret key $d_{ID} = \mathsf{KeyDer}(msk, ID)$, and it returns the session key $K = e(d_{ID}, C)$.

In the following theorem we show that the above scheme satisfies the pseudo-random decapsulation property assuming that the Decisional Bilinear Diffie–Hellman assumption (DBDH) is hard. In particular, for this scheme we prove pseudo-random decapsulation w.r.t. any arbitrary distribution $\mathcal{D}_{\mathcal{ID}}$ over the identity space (i.e., the proof works for any $ID_0 \in \mathcal{ID}$). The proof is inspired to the one given in [7] (Lemma 4.2).

Theorem 12. *If the DBDH assumption holds in $\mathbb{G}, \mathbb{G}_T$ and H_1 is modeled as a random oracle, then the scheme $\mathcal{BFKEM}$ satisfies pseudo-random decapsulation.*

Proof. More formally, let $\mathcal{A}$ be an adversary for the pseudo-random decapsulation (IB-KEM-RDECAP) of $\mathcal{BFKEM}$ with advantage $\mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{A},\mathcal{BFKEM}}(k) = \epsilon(k)$. Let $H_1 : \{0,1\}^* \to \mathbb{G}$ be a random oracle. Thus, assuming q_E to be an upper bound to the number of key derivations asked to the oracle, we show how to construct a simulator $\mathcal{B}$ that uses $\mathcal{A}$ to solve DBDH with advantage at least $\epsilon(k)/e(1+q_E)$.

$\mathcal{B}$ receives from its challenger a DBDH tuple $(P, P_1, P_2, P_3, Z) = (P, P^a, P^b, P^c, Z)$ where $P \in \mathbb{G}$ is a random generator and a, b, c are taken at random in $\mathbb{Z}_p$. The challenger flips a binary coin $\nu \in \{0,1\}$. If $\nu = 0$ it sets $Z = e(P,P)^{abc}$, otherwise it picks a random $Z \xleftarrow{\$} \mathbb{G}_T$. $\mathcal{B}$ must output 0 if it believes that $Z = e(P,P)^{abc}$, and 1 otherwise.

First, $\mathcal{B}$ constructs the public key for the $\mathcal{BFKEM}$ scheme by setting $mpk = (p, \mathbb{G}, \mathbb{G}_T, P, P_{pub} = P_1, H_1)$. It also sets $C_0 = P_3$ and gives (mpk, C_0) to $\mathcal{A}_1$.[3] The simulator controls the random oracle H_1 as follows. It maintains a list H_1^{list} of tuples $(ID_i, Q_i, \beta_i, coin_i)$. When the adversary queries H_1 on input ID, $\mathcal{B}$ checks if $ID \in H_1^{list}$. If this is the case and $(ID, Q, \beta, coin)$ is the correlated tuple, then it outputs $H_1(ID) = Q$. Otherwise $\mathcal{B}$ flips a random $coin \in \{0,1\}$ such that $\Pr[coin = 0] = \delta$. It picks a random $\beta \xleftarrow{\$} \mathbb{Z}_p^*$. If $coin = 0$ it sets $Q = P^\beta$, otherwise it sets $Q = P_2^\beta$.

The adversary is allowed to make key derivation queries to the oracle $\mathsf{KeyDer}(\cdot)$. $\mathcal{B}$ answers such queries in the following way. On input ID_i it queries $H_1(ID_i)$. Let $(ID_i, Q_i, \beta_i, coin_i)$ the tuple in H_1^{list}. If $coin_i = 1$ the simulator fails and outputs a random guess $\nu' \in \{0,1\}$. Otherwise $\mathcal{B}$ sets $d_{ID} = P_1^{\beta_i}$. In the end of this phase $\mathcal{A}_1$ outputs a challenge identity $\overline{ID}$. $\mathcal{B}$ runs $H_1(\overline{ID})$ to obtain $(\overline{ID}, \overline{Q}, \overline{\beta}, \overline{coin})$. If $\overline{coin} = 0$ $\mathcal{B}$ fails and outputs a random guess. Otherwise it sets $\overline{K} = Z^{\overline{\beta}}$ and gives $\overline{K}$ to $\mathcal{A}_2$. When the adversary outputs its guess ν' for $\overline{K}$ $\mathcal{B}$ outputs the same value to its challenger.

We observe that if $Z = e(P,P)^{abc}$ then $\overline{K} = e(P,P)^{abc\overline{\beta}} = e(d_{\overline{ID}}, C_0)$ is a correct session key obtained by decapsulating C_0 with identity $\overline{ID}$. Otherwise if Z is random, $\overline{K}$

[3] We observe that in this case it is not necessary to explicitly choose an identity ID_0.

will be random as well. Let fail be the event that $\mathcal{B}$ aborts during the simulation. Clearly, if $\mathcal{B}$ does not abort the simulation is perfect. Thus we have:

$$\mathbf{Adv}^{\text{DBDH}}_{\mathcal{B}}(k) = \mathbf{Adv}^{\text{IB-KEM-RDECAP}}_{\mathcal{A},\mathcal{BFKEM}}(k)\,\Pr[\overline{\mathsf{fail}}].$$

If q_E is the number of key derivation queries issued to the oracle, then the probability that $\mathcal{B}$ does not abort is $\delta^{q_E}(1-\delta)$. This value is maximized with $\delta_{opt} = 1 - \frac{1}{q_E+1}$ for which we obtain $\Pr[\overline{\mathsf{fail}}] \geq \frac{1}{e(q_E+1)}$. □

Appendix B. The VRF by Dodis and Yampolskiy

In this section we recall the VRF by Dodis and Yampolskiy [22].

- $\mathsf{Gen}(1^k)$: The key generation algorithm runs $\mathcal{G}(1^k)$ to obtain the description of the groups G, G_T and of a bilinear map $e : G \times G \to G_T$. The description of G contains a generator $g \in G$. Then the algorithm picks a random $s \xleftarrow{\$} \mathbb{Z}_p$ and sets $h = g^s$, $vpk = (g, h)$, $vsk = s$.
- $\mathsf{Func}_{vsk}(x)$: Let $\mathsf{Func}_{vsk}(x) = (F_{vsk}(x), \pi_{vsk}(x))$. One sets $\mathsf{Func}_{vsk}(x) = e(g,g)^{1/(s+x)}$ as the VRF output and $\pi_{vsk}(x) = g^{1/(s+x)}$ as the proof of correctness.
- $\mathsf{V}(vpk, x, y, \pi_x)$: To verify if y was computed correctly, one checks that $e(g^x \cdot h, \pi_x) = e(g,g)$ and $y = e(g, \pi_x)$.

Appendix C. Proof of Theorem 10

The unique decapsulation property was already shown in Theorem 8 in Sect. 5.2. Hence, we only prove that the pseudo-random decapsulation holds under the n-DDHE assumption. As in the selective case, we prove pseudo-random decapsulation w.r.t. any arbitrary distribution $\mathcal{D}_{\mathcal{ID}}$ (i.e., the proof works for any $ID_0 \in \mathcal{ID}$). As stated before, this proof follows very closely the proof of security of the Hohenberger–Waters VRF [32] and is only provided here for completeness.

For the sake of contradiction, assume there exists an efficient adversary $\mathcal{A}$ that is able to break the pseudo-random decapsulation property of our scheme with non-negligible advantage ϵ. Then we will show that we can build an algorithm $\mathcal{B}$ out of $\mathcal{A}$ to break the n-DDHE assumption with advantage at least $\frac{3\epsilon}{64Q(\ell+1)}$, where Q is the number of secret key queries made by $\mathcal{A}$ during the game and $n = 4Q(\ell+1)+1$. Recall that ℓ is the length in bits of the identities in our scheme.

We first give a description of the algorithm $\mathcal{B}$ and then we will describe two hybrid games where we slightly change the original experiment to ease our analysis.

$\mathcal{B}$ gets as input a tuple $(g, h, g^b, g^{b^2}, \ldots, g^{b^{n-1}}, g^{b^{n+1}}, \ldots, g^{b^{2n}}, Z)$ and proceeds as follows.

It sets $m = 4Q$ and then chooses an integer $k \xleftarrow{\$} \{0, \ldots, \ell\}$ and other random values $r_{i,b} \xleftarrow{\$} \{0, \ldots, m-1\}$ for $i = 1$ to ℓ and $b \in \{0,1\}$. Then it picks random exponents $a', \alpha'_i, \beta'_i \xleftarrow{\$} \mathbb{Z}^*_p$ and an identity $ID_0 \leftarrow \mathcal{D}_{\mathcal{ID}}$, and sets:

$$C_0 = h, \qquad g_1 = \left(g^{b^{m(k+1)}}\right)^{a'}, \qquad g_{0,i} = \left(g^{b^{r_{i,0}}}\right)^{\beta'_i} g_{1,i} = \left(g^{b^{r_{i,1}}}\right)^{\alpha'_i}$$

for $i = 1$ to ℓ. Finally, it gives $mpk = (g, g_1, \{g_{0,i}, g_{1,i}\}_{i=1}^{\ell})$, C_0 and ID_0 to $\mathcal{A}$.

Before describing the rest of the proof, we show that the public key can be computed by $\mathcal{B}$. To do this, we evaluate an upper bound on the exponents of the values b contained in the public key elements. Note that the secret key is implicitly set as $a = b^{m(k+1)}a'$, $\alpha_i = b^{r_{1,i}}\alpha_i'$, $\beta_i = b^{r_{0,i}}\beta_i'$, and it is easy to check that: $m(k+1) \leq 4Q(\ell+1) < n$, $r_{b,i} \leq m-1 < n$, for all $b \in \{0,1\}$ and $i = 1, \ldots, \ell$. Therefore, all the elements of the public key can be efficiently computed by $\mathcal{B}$. Moreover, it is easy to observe that *mpk* is correctly distributed.

Similar bounds will be useful to check when our simulator $\mathcal{B}$ will be able to compute a secret key for an identity *ID*. To this end, we define the following functions:

$$C(ID) = m(k+1) + \sum_{i=1}^{\ell} r_{i,ID_i}, \qquad \hat{C}(ID, j) = \sum_{i=1}^{j} r_{i,ID_i},$$

$$J(ID) = a' \prod_{i=1}^{\ell} (\alpha_i')^{ID_i} (\beta_i')^{(1-ID_i)}, \qquad \hat{J}(ID, j) = a' \prod_{i=1}^{j} (\alpha_i')^{ID_i} (\beta_i')^{(1-ID_i)}.$$

Then, observe that for any identity *ID* the following bounds hold:

1. $C(ID) \leq m(\ell+1) + (m-1)\ell = 2m\ell + m - \ell < 2n$,
2. $\forall j \in \{1, \ldots, \ell\}$: $\hat{C}(ID, j) \leq (m-1)\ell < n$.

Now let us go back to the simulation. In order to answer secret key queries, $\mathcal{B}$ proceeds as follows. Let *ID* be the queried identity. If $C(ID) = n$, then it aborts and outputs a random bit. Otherwise, it computes the auxiliary information and the secret key as follows:

$$h_j = g^{b^{\hat{C}(ID,j)}\hat{J}(ID,j)}, \quad \forall j = 1, \ldots, \ell \quad \text{and} \quad sk_{ID} = g^{b^{C(ID)}J(ID)}.$$

Given the bounds shown above, one can verify that $\mathcal{B}$ is always able to compute the auxiliary information $aux_{ID} = (h_1, \ldots, h_\ell)$. In contrast, $\mathcal{B}$ can compute the secret key part sk_{ID} as long as $C(ID) \neq n$.

Finally, when $\mathcal{A}$ outputs the challenge identity $\overline{ID}$, $\mathcal{B}$ checks whether $C(\overline{ID}) = n$. If $C(\overline{ID}) \neq n$, then it aborts and outputs a random bit. Otherwise, it computes the session key $K = Z^{J(\overline{ID})}$, and gives it to $\mathcal{A}$. When $\mathcal{A}$ outputs a guess c', $\mathcal{B}$ outputs the same bit.

If Z is a random value in G_T, then K is a random session key. Otherwise, if $Z = e(g,h)^{b^n}$, then one can verify that K is correctly distributed as a real decapsulation of C_0 under identity $\overline{ID}$, namely $K = e(sk_{\overline{ID}}, C_0) = e(g^{b^{C(\overline{ID})}J(\overline{ID})}, C_0)$.

If $\mathcal{A}$ returns the correct answer, then $\mathcal{B}$ will be successful. However, as one can notice, the game presented by $\mathcal{B}$ to $\mathcal{A}$ is slightly different from the real pseudo-random decapsulation experiment as it contains several points in which $\mathcal{B}$ might stop the simulation. In particular, the abort condition depends on the set of queries made by $\mathcal{A}$.

Therefore, in order to analyze $\mathcal{A}$'s advantage, we define the following hybrid games and then show that $\mathcal{A}$'s advantage in these games cannot change too much. We stress that this analysis is essentially the same as that provided by Hohenberger and Waters for their scheme [33]. We give it here for completeness.

Game 0 is the same as the real pseudo-random decapsulation experiment.

Game 1 is the same as Game 0 except that we add an abort condition at the end of the game. Namely, the adversary $\mathcal{A}$ is first run in an experiment identical to pseudo-random decapsulation. Next, after $\mathcal{A}$ returns its output b', the challenger proceeds as follows. Let $\overrightarrow{ID} = (ID_1, \ldots, ID_Q, \overline{ID})$ be the set of identities queried by the adversary and let k be randomly chosen in $\{0, \ldots, \ell\}$ and $\overrightarrow{r} = (r_{1,0}, r_{1,1}, \ldots, r_{\ell,0}, r_{\ell,1})$ be a vector of values chosen at random in $\{0, \ldots, m-1\}$, where $m = 4Q$. For $\overrightarrow{ID}$, $\overrightarrow{r}$ and k, we define the following functions:

$$K(ID) = \begin{cases} 0 & \text{if } \sum_{i=1}^{\ell} r_{i,ID_i} \equiv 1 \bmod m; \\ 1 & \text{otherwise,} \end{cases}$$

$$\tau(\overrightarrow{ID}, \overrightarrow{r}, k) = \begin{cases} 1 & \text{if } \sum_{i=1}^{\ell} r_{i,ID*_i} \neq m(\ell - k) + 1 \bigvee_{i=1}^{Q} K(ID_i) = 0; \\ 0 & \text{otherwise.} \end{cases}$$

Given the random choices $\overrightarrow{r}$ and k, the function τ tells whether a set of identities $\overrightarrow{ID}$ asked by the adversary causes an abort or not. More precisely, τ evaluates to 0 if $\overrightarrow{ID}$ does not cause an abort w.r.t. $\overrightarrow{r}$ and k, and 1 otherwise.

After the adversary $\mathcal{A}$ is done the challenger evaluates an approximation p' for the probability $p(\overrightarrow{ID}) = \Pr[\tau(\overrightarrow{ID}, \overrightarrow{r}, k) = 0]$ by repeating $O(\epsilon^{-2} \ln(\epsilon^{-1} p_{\min}^{-1} \ln(p_{\min^{-1}})))$ times the following task: sample fresh random $\overrightarrow{r'}, k'$ values and evaluate $\tau(\overrightarrow{ID}, \overrightarrow{r'}, k')$. Let $p_{\min} = \frac{1}{8Q(\ell+1)}$ (as established in Claim 1 below). If $\tau(\overrightarrow{ID}, \overrightarrow{r}, k) = 1$, then the challenger outputs a random bit (discarding the bit b' returned by $\mathcal{A}$). But, even if $\tau(\overrightarrow{ID}, \overrightarrow{r}, k) = 0$, the challenger outputs a random bit with probability $1 - \frac{p_{\min}}{p'}$. Otherwise, it outputs b'.

Game 2 is the same as Game 1 except that the challenger checks the abort condition at every new query received from $\mathcal{A}$. If the conditioned is satisfied, then the challenger aborts immediately (without waiting for the termination of $\mathcal{A}$) by returning a random bit. However, even if no abort condition was satisfied during the run of $\mathcal{A}$, the challenger executes the final artificial abort phase exactly as in Game 1.

In order to complete the proof, we will show that if $\mathcal{A}$ wins in Game 0 with probability $\frac{1}{2} + \epsilon$, then it will have success in Game 2 with probability $\geq \frac{1}{2} + \frac{3\epsilon}{64Q(\ell+1)}$, from which it turns out that $\mathcal{B}$ gains advantage $\mathbf{Adv}_{\mathcal{B}}^{\mathrm{DDHE}} \geq \frac{3\epsilon}{64Q(\ell+1)}$.

To prove this fact, we show the following claims.

Claim 1. *For any set $\overrightarrow{ID}$ of identities, it holds $p(\overrightarrow{ID}) = \Pr[\tau(\overrightarrow{ID}, \overrightarrow{r}, k) = 0] \geq p_{\min} = \frac{1}{8Q(\ell+1)}$.*

Proof. This claim aims at bounding the probability, over the random choices $\overrightarrow{r}$ and k, that a given set of identities $\overrightarrow{ID}$ does not cause abort. Assuming that the adversary makes

at most Q key derivation queries, we obtain that the probability $\Pr[\tau(\overrightarrow{ID}, \overrightarrow{r}, k) = 0]$ is

$$= \Pr\left[\bigwedge_{i=1}^{Q} K(ID_i) = 1 \wedge \sum_{i=1}^{\ell} r_{i,\overline{ID}_i} = m(\ell - k) + 1\right] \tag{C.1}$$

$$= \left(1 - \Pr\left[\bigvee_{i=1}^{Q} K(ID_i) = 0\right]\right) \Pr\left[\sum_{i=1}^{\ell} r_{i,\overline{ID}_i} = m(\ell - k) + 1 | \bigwedge_{i=1}^{Q} K(ID_i) = 1\right] \tag{C.2}$$

$$\geq \left(1 - \sum_{i=1}^{Q} \Pr\big[K(ID_i) = 0\big]\right) \Pr\left[\sum_{i=1}^{\ell} r_{i,\overline{ID}_i} = m(\ell - k) + 1 | \bigwedge_{i=1}^{Q} K(ID_i) = 1\right] \tag{C.3}$$

$$= \left(1 - \frac{Q}{m}\right) \Pr\left[\sum_{i=1}^{\ell} r_{i,\overline{ID}_i} = m(\ell - k) + 1 | \bigwedge_{i=1}^{Q} K(ID_i) = 1\right] \tag{C.4}$$

$$= \left(1 - \frac{Q}{m}\right) \frac{1}{\ell + 1} \Pr\left[K(\overline{ID}) = 0 | \bigwedge_{i=1}^{Q} K(ID_i) = 1\right] \tag{C.5}$$

$$= \left(1 - \frac{Q}{m}\right) \frac{1}{\ell + 1} \frac{\Pr[K(\overline{ID}) = 0] \Pr[\bigwedge_{i=1}^{Q} K(ID_i) = 1 | K(\overline{ID}) = 0]}{\Pr[\bigwedge_{i=1}^{Q} K(ID_i) = 1]} \tag{C.6}$$

$$\geq \left(1 - \frac{Q}{m}\right) \frac{1}{(\ell + 1)m} \Pr\left[\bigwedge_{i=1}^{Q} K(ID_i) = 1 | K(\overline{ID}) = 0\right] \tag{C.7}$$

$$= \left(1 - \frac{Q}{m}\right) \frac{1}{(\ell + 1)m} \left(1 - \Pr\left[\bigvee_{i=1}^{Q} K(ID_i) = 0 | K(\overline{ID}) = 0\right]\right) \tag{C.8}$$

$$\geq \left(1 - \frac{Q}{m}\right) \frac{1}{(\ell + 1)m} \left(1 - \sum_{i=1}^{Q} \Pr\big[K(ID_i) = 0 | K(\overline{ID}) = 0\big]\right) \tag{C.9}$$

$$= \left(1 - \frac{Q}{m}\right)^2 \frac{1}{(\ell + 1)m} \tag{C.10}$$

$$\geq \left(1 - \frac{2Q}{m}\right) \frac{1}{(\ell + 1)m} \tag{C.11}$$

$$= \frac{1}{8Q(\ell + 1)}. \tag{C.12}$$

To see how some of the above equations are obtained, we make the following observations. Equations (C.4) and (C.7) derive from the fact that $\Pr[K(ID) = 0] = 1/m$ for any ID. The factor $\frac{1}{\ell+1}$ in Equation (C.5) is the probability that the simulator hits the right k. Equation (C.10) derives from the pairwise independence of the probability that $K(\cdot) = 0$, namely $\Pr[K(ID) = 0] = \Pr[K(ID') = 0]$ for all $ID \neq ID'$. Indeed, notice that if two identities are different, then the sums in $K(ID)$ and $K(ID')$ will contain at least

two different values $r_{j,0}, r_{j,1}$ at the index j where ID and ID' differ. Finally, the last equation comes from setting $m = 4Q$. □

Claim 2. *For any set $\overrightarrow{ID}$ of identities, the probability that Game* 1 aborts *is at least* $1 - p_{\min} - p_{\min}\frac{3}{8}\epsilon$.

Proof. Recall that in Game 1 the probability that the vector $\overrightarrow{ID}$ provided by the adversary does not cause an abort is approximated with p' by taking $T = O(\epsilon^{-2}\ln(\epsilon^{-1})p_{\min}^{-1}\ln(p_{\min}^{-1}))$ samples. By Chernoff bound we have that

$$\Pr\left[T \cdot p' < T \cdot p(\overrightarrow{ID})\left(1 - \frac{\epsilon}{8}\right)\right] < e^{-(128\epsilon^{-2}\ln((\epsilon/8)^{-1})p_{\min}^{-1}\ln(p_{\min}^{-1})p_{\min}(\epsilon/8)^2/2)}$$

holds for any $\overrightarrow{ID}$. This can be simply reduced to

$$\Pr\left[p' < p(\overrightarrow{ID})\left(1 - \frac{\epsilon}{8}\right)\right] < p_{\min}\frac{\epsilon}{8}.$$

Now, recall that Game 1 artificially aborts with probability $1 - \frac{p_{\min}}{p'}$. Therefore, if we measure the probability of aborting in Game 1, we obtain that

$$\begin{aligned}
\Pr[abort] &= 1 - \Pr[\neg abort] = 1 - \Pr[\neg \textit{Regular Abort}]\Pr[\neg \textit{Arti ficial Abort}] \\
&= 1 - p(\overrightarrow{ID})\Pr[\neg \textit{Arti ficial Abort}] \\
&\geq 1 - p(\overrightarrow{ID})\left(p_{\min}\frac{\epsilon}{8} + \frac{p_{\min}}{p(\overrightarrow{ID})(1-\epsilon/8)}\right) \\
&\geq 1 - \left(p_{\min}\frac{\epsilon}{8} + \frac{p_{\min}}{(1-\epsilon/8)}\right) \\
&\geq 1 - \left(\frac{p_{\min}\epsilon}{8} + p_{\min}\left(1 + \frac{2\epsilon}{8}\right)\right) \\
&\geq 1 - p_{\min} - p_{\min}\frac{3\epsilon}{8}.
\end{aligned}$$

□

Claim 3. *For any set $\overrightarrow{ID}$ of identities, the probability that Game* 1 *does* not *abort is at least* $p_{\min} - p_{\min}\frac{1}{4}\epsilon$.

Proof. As in the previous claim, let $T = O(\epsilon^{-2}\ln(\epsilon^{-1})p_{\min}^{-1}\ln(p_{\min}^{-1}))$ be the number of samples that are used to estimate the approximation p' of $p(\overrightarrow{ID})$. By Chernoff bounds, we have that

$$\Pr\left[T \cdot p' > T \cdot p(\overrightarrow{ID})\left(1 + \frac{\epsilon}{8}\right)\right] < e^{-(256\epsilon^{-2}\ln((\epsilon/8)^{-1})p_{\min}^{-1}\ln(p_{\min}^{-1})p_{\min}(\epsilon/8)^2/4}$$

holds for any set $\overrightarrow{ID}$. This can be reduced to

$$\Pr\left[p' > p(\overrightarrow{ID})\left(1 + \frac{\epsilon}{8}\right)\right] < p_{\min}\frac{\epsilon}{8}.$$

Now, recall that the probability that Game 1 does not abort is equal to the probability that neither a regular abort or an artificial abort occur during the simulation. Given an estimation p', an artificial abort does not happen with probability $p_{\min}/p'$, therefore the probability of not aborting in Game 1 is

$$\begin{aligned}
\Pr[\neg abort] &= \Pr[\neg \textit{Regular Abort}]\Pr[\neg \textit{Arti ficial Abort}] \\
&\geq p(\overrightarrow{ID})\left(1-\frac{p_{\min}\epsilon}{8}\right)\left(\frac{p_{\min}}{p(\overrightarrow{ID})(1+\epsilon/8)}\right) \\
&\geq p_{\min}\left(1-\frac{\epsilon}{8}\right)^2 \\
&\geq p_{\min}\left(1-\frac{1}{4}\epsilon\right).
\end{aligned}$$

□

Claim 4. *If the adversary $\mathcal{A}$ wins in Game 0 with probability $\frac{1}{2}+\epsilon$, then it wins in Game 1 with probability $\geq \frac{1}{2}+\frac{3\epsilon}{64Q(\ell+1)}$.*

Proof. The probability that an adversary wins Game 1 is equal to the probability that such experiment outputs 1, which is:

$$\begin{aligned}
\Pr[G_1=1] &= \Pr[G_1=1|abort]\Pr[abort]+\Pr[G_1=1|\neg abort]\Pr[\neg abort] \\
&= \frac{1}{2}\Pr[abort]+\Pr\left[b'=b|\neg abort\right]\Pr[\neg abort] \\
&= \frac{1}{2}\Pr[abort]+\Pr[b'=b]\Pr\left[\neg abort|b'=b\right] \\
&= \frac{1}{2}\Pr[abort]+\left(\frac{1}{2}+\epsilon\right)\Pr\left[\neg abort|b'=b\right] \\
&\geq \frac{1}{2}\left(1-p_{\min}-p_{\min}\frac{3\epsilon}{8}\right)+\left(\frac{1}{2}+\epsilon\right)\left(p_{\min}-p_{\min}\frac{\epsilon}{4}\right) \\
&\geq \frac{1}{2}+\epsilon p_{\min}-p_{\min}\frac{\epsilon}{4}-p_{\min}\frac{3\epsilon}{8} \\
&= \frac{1}{2}+\frac{3}{8}\epsilon p_{\min} \\
&= \frac{1}{2}+\frac{3\epsilon}{64Q(\ell+1)}.
\end{aligned}$$

The equations above are obtained by standard probability arguments and by using the results of the previous claims. □

Claim 5. *The adversary $\mathcal{A}$ has the same success probability in Game 1 and Game 2.*

To see the proof of this claim, observe that the only difference between the two games is the time in which the abort condition is issued and the game terminates. However, it is easy to see that such a change does not modify the output distribution of the experiment.

Finally, to complete the proof of Theorem 10, we simply observe that the view of the adversary $\mathcal{A}$ in Game 2 is identical to the view that $\mathcal{A}$ obtains in the simulation provided by our algorithm $\mathcal{B}$.

References

[1] M. Abdalla, D. Catalano, D. Fiore, Verifiable random functions from identity-based key encapsulation, in *Advances in Cryptology—EUROCRYPT 2009*, Cologne, Germany, April 26–30, ed. by A. Joux. Lecture Notes in Computer Science, vol. 5479 (Springer, Berlin, 2009), pp. 554–571

[2] B. Barak, Y. Dodis, H. Krawczyk, O. Pereira, K. Pietrzak, F.-X. Standaert, Y. Yu, Leftover hash lemma, revisited, in *Advances in Cryptology—CRYPTO 2011*, Santa Barbara, CA, USA, August. Lecture Notes in Computer Science (Springer, Berlin, 2011), pp. 1–20

[3] K. Bentahar, P. Farshim, J. Malone-Lee, N.P. Smart, Generic constructions of identity-based and certificateless KEMs. *J. Cryptol.* **21**(2), 178–199 (2008)

[4] D. Boneh, X. Boyen, Efficient selective-ID secure identity based encryption without random oracles, in *Advances in Cryptology—EUROCRYPT 2004*, Interlaken, Switzerland, May 2–6, ed. by C. Cachin, J. Camenisch. Lecture Notes in Computer Science, vol. 3027 (Springer, Berlin, 2004), pp. 223–238

[5] D. Boneh, X. Boyen, Secure identity based encryption without random oracles, in *Advances in Cryptology—CRYPTO 2004*, Santa Barbara, CA, USA, August 15–19, ed. by M. Franklin. Lecture Notes in Computer Science, vol. 3152 (Springer, Berlin, 2004), pp. 443–459

[6] D. Boneh, X. Boyen, Short signatures without random oracles, in *Advances in Cryptology—EUROCRYPT 2004*, Interlaken, Switzerland, May 2–6, ed. by C. Cachin, J. Camenisch. Lecture Notes in Computer Science, vol. 3027 (Springer, Berlin, 2004), pp. 56–73

[7] D. Boneh, M.K. Franklin, Identity-based encryption from the Weil pairing, in *Advances in Cryptology—CRYPTO 2001*, Santa Barbara, CA, USA, August 19–23, ed. by J. Kilian. Lecture Notes in Computer Science, vol. 2139 (Springer, Berlin, 2001), pp. 213–229

[8] D. Boneh, X. Boyen, H. Shacham, Short group signatures, in *Advances in Cryptology—CRYPTO 2004*, Santa Barbara, CA, USA, August 15–19, ed. by M. Franklin. Lecture Notes in Computer Science, vol. 3152 (Springer, Berlin, 2004), pp. 41–55

[9] D. Boneh, X. Boyen, E.-J. Goh, Hierarchical identity based encryption with constant size ciphertext, in *Advances in Cryptology—EUROCRYPT 2005*, Aarhus, Denmark, May 22–26, ed. by R. Cramer. Lecture Notes in Computer Science, vol. 3494 (Springer, Berlin, 2005), pp. 440–456

[10] D. Boneh, C. Gentry, B. Waters, Collusion resistant broadcast encryption with short ciphertexts and private keys, in *Advances in Cryptology—CRYPTO 2005*, Santa Barbara, CA, USA, August 14–18, ed. by V. Shoup. Lecture Notes in Computer Science, vol. 3621 (Springer, Berlin, 2005), pp. 258–275

[11] D. Boneh, H.W. Montgomery, A. Raghunathan, Algebraic pseudorandom functions with improved efficiency from the augmented cascade, in *ACM CCS 10: The 17th Conference on Computer and Communications Security*, Chicago, Illinois, USA, October 4–8, ed. by E. Al-Shaer, A.D. Keromytis, V. Shmatikov (ACM Press, New York, 2010), pp. 4–8

[12] Z. Brakerski, S. Goldwasser, G.N. Rothblum, V. Vaikuntanathan, Weak verifiable random functions, in *TCC 2009: The 6th Theory of Cryptography Conference*, March 15–17, ed. by R. Omer. Lecture Notes in Computer Science, vol. 5444 (Springer, Berlin, 2009), pp. 558–576

[13] R. Canetti, S. Halevi, J. Katz, A forward-secure public-key encryption scheme, in *Advances in Cryptology—EUROCRYPT 2003*, Warsaw, Poland, May 4–8, ed. by E. Biham. Lecture Notes in Computer Science, vol. 2656 (Springer, Berlin, 2003), pp. 255–271

[14] D. Cash, D. Hofheinz, E. Kiltz, C. Peikert, Bonsai trees, or how to delegate a lattice basis, in *Advances in Cryptology—EUROCRYPT 2010*, French Riviera, May 30–June 3, ed. by H. Gilbert. Lecture Notes in Computer Science, vol. 6110 (Springer, Berlin, 2010), pp. 523–552

[15] D. Cash, D. Hofheinz, E. Kiltz, C. Peikert, Bonsai trees, or how to delegate a lattice basis. *J. Cryptol.* **25**, 601–639 (2012)

[16] M. Chase, A. Lysyanskaya, Simulatable VRFs with applications to multi-theorem NIZK, in *Advances in Cryptology—CRYPTO 2007*, Santa Barbara, CA, USA, August 19–23, ed. by A. Menezes. Lecture Notes in Computer Science, vol. 4622 (Springer, Berlin, 2007), pp. 303–322

[17] J.H. Cheon, Security analysis of the strong Diffie–Hellman problem, in *Advances in Cryptology—EUROCRYPT 2006*, St. Petersburg, Russia, May 28–June 1, ed. by S. Vaudenay. Lecture Notes in Computer Science, vol. 4004 (Springer, Berlin, 2006), pp. 1–11

[18] R. Cramer, G. Hanaoka, D. Hofheinz, H. Imai, E. Kiltz, R. Pass, A. Shelat, V. Vaikuntanathan, Bounded CCA2-secure encryption, in *Advances in Cryptology—ASIACRYPT 2007*, Kuching, Malaysia, December 2–6, ed. by K. Kurosawa. Lecture Notes in Computer Science, vol. 4833 (Springer, Berlin, 2007), pp. 502–518

[19] W. Diffie, M.E. Hellman, New directions in cryptography. *IEEE Trans. Inf. Theory* **22**(6), 644–654 (1976)

[20] Y. Dodis, Efficient construction of (distributed) verifiable random functions, in *PKC 2003: The 6th International Workshop on Theory and Practice in Public Key Cryptography*, Miami, USA, January 6–8, ed. by Y. Desmedt. Lecture Notes in Computer Science, vol. 2567 (Springer, Berlin, 2003), pp. 1–17

[21] Y. Dodis, P. Puniya, Verifiable random permutations. Cryptology ePrint Archive, Report 2006/078, 2006. http://eprint.iacr.org/

[22] Y. Dodis, A. Yampolskiy, A verifiable random function with short proofs and keys, in *PKC 2005: the 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets*, Les Diablerets, Switzerland, January 23–26, ed. by S. Vaudenay. Lecture Notes in Computer Science, vol. 3386 (Springer, Berlin, 2005), pp. 416–431

[23] Y. Dodis, J. Katz, S. Xu, M. Yung, Key-insulated public key cryptosystems, in *Advances in Cryptology—EUROCRYPT 2002*, Amsterdam, The Netherlands, April 28–May 2, ed. by L.R. Knudsen. Lecture Notes in Computer Science, vol. 2332 (Springer, Berlin, 2002), pp. 65–82

[24] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in *Advances in Cryptology—CRYPTO'84*, Santa Barbara, CA, USA, August 19–23, ed. by G.R. Blakley, D. Chaum. Lecture Notes in Computer Science, vol. 196 (Springer, Berlin, 1985), pp. 10–18

[25] P. Erdös, P. Frankel, Z. Furedi, Families of finite sets in which no set is covered by the union of r others. *Isr. J. Math.* **51**, 79–89 (1985)

[26] D. Fiore, D. Schröder, Uniqueness is a different story: impossibility of verifiable random functions from trapdoor permutations, in *TCC 2012: The 9th Theory of Cryptography Conference*, Taormina, Sicily, Italy, March 19–21, ed. by R. Cramer. Lecture Notes in Computer Science, vol. 7194 (Springer, Berlin, 2012), pp. 636–653

[27] C. Gentry, Practical identity-based encryption without random oracles, in *Advances in Cryptology—EUROCRYPT 2006*, St. Petersburg, Russia, May 28–June 1, ed. by S. Vaudenay. Lecture Notes in Computer Science, vol. 4004 (Springer, Berlin, 2006), pp. 445–464

[28] O. Goldreich, L.A. Levin, A hard-core predicate for all one-way functions, in *21st ACM STOC Annual ACM Symposium on Theory of Computing*, Seattle, Washington, USA, May 15–17 (ACM Press, New York, 1989), pp. 25–32

[29] S. Goldwasser, R. Ostrovsky, Invariant signatures and non-interactive zero-knowledge proofs are equivalent (extended abstract), in *Advances in Cryptology—CRYPTO'92*, Santa Barbara, CA, USA, August 16–20, ed. by E.F. Brickell. Lecture Notes in Computer Science, vol. 740 (Springer, Berlin, 1993), pp. 228–245

[30] J. Håstad, R. Impagliazzo, L.A. Levin, M. Luby, A pseudorandom generator from any one-way function. *SIAM J. Comput.* **28**(4), 1364–1396 (1999)

[31] S.-H. Heng, K. Kurosawa, k-resilient identity-based encryption in the standard model, in *Topics in Cryptology—CT-RSA 2004*, San Francisco, CA, USA, February 23–27, ed. by T. Okamoto. Lecture Notes in Computer Science, vol. 2964 (Springer, Berlin, 2004), pp. 67–80

[32] S. Hohenberger, B. Waters, Realizing hash-and-sign signatures under standard assumptions, in *Advances in Cryptology—EUROCRYPT 2009*, Cologne, Germany, April 26–30, ed. by A. Joux. Lecture Notes in Computer Science, vol. 5479 (Springer, Berlin, 2009), pp. 333–350

[33] S. Hohenberger, B. Waters, Constructing verifiable random functions with large input spaces, in *Advances in Cryptology—EUROCRYPT 2010*, French Riviera, May 30–June 3, ed. by H. Gilbert. Lecture Notes in Computer Science, vol. 6110 (Springer, Berlin, 2010), pp. 656–672

[34] S. Jarecki, V. Shmatikov, Handcuffing big brother: an abuse-resilient transaction escrow scheme, in *Advances in Cryptology—EUROCRYPT 2004*, Interlaken, Switzerland, May 2–6, ed. by C. Cachin, J. Camenisch. Lecture Notes in Computer Science, vol. 3027 (Springer, Berlin, 2004), pp. 590–608

[35] R. Kumar, S. Rajagopalan, A. Sahai, Coding constructions for blacklisting problems without computational assumptions, in *Advances in Cryptology—CRYPTO'99*, Santa Barbara, CA, USA, August 15–19, ed. by M.J. Wiener. Lecture Notes in Computer Science, vol. 1666 (Springer, Berlin, 1999), pp. 609–623

[36] M. Liskov, Updatable zero-knowledge databases, in *Advances in Cryptology—ASIACRYPT 2005*, Chennai, India, December 4–8, ed. by B.K. Roy. Lecture Notes in Computer Science, vol. 3788 (Springer, Berlin, 2005), pp. 174–198

[37] M. Luby, C. Rackoff, How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.* **17**(2) (1988)

[38] A. Lysyanskaya, Unique signatures and verifiable random functions from the DH-DDH separation, in *Advances in Cryptology—CRYPTO 2002*, Santa Barbara, CA, USA, August 18–22, ed. by M. Yung. Lecture Notes in Computer Science, vol. 2442 (Springer, Berlin, 2002), pp. 597–612

[39] S. Micali, L. Reyzin, Soundness in the public-key model, in *Advances in Cryptology—CRYPTO 2001*, Santa Barbara, CA, USA, August 19–23, ed. by J. Kilian. Lecture Notes in Computer Science, vol. 2139 (Springer, Berlin, 2001), pp. 542–565

[40] S. Micali, R.L. Rivest, Micropayments revisited, in *Topics in Cryptology—CT-RSA 2002*, San Jose, CA, USA, February 18–22, ed. by B. Preneel. Lecture Notes in Computer Science, vol. 2271 (Springer, Berlin, 2002), pp. 149–163

[41] S. Micali, M.O. Rabin, S.P. Vadhan, Verifiable random functions, in *40th Annual Symposium on Foundations of Computer Science*, New York, New York, USA, October 17–19 (IEEE Computer Society Press, Los Alamitos, 1999), pp. 120–130

[42] M. Naor, O. Reingold, Number-theoretic constructions of efficient pseudo-random functions, in *38th Annual Symposium on Foundations of Computer Science*, Miami Beach, Florida, October 19–22 (IEEE Computer Society Press, Los Alamitos, 1997), pp. 458–467

[43] R. Sakai, M. Kasahara, Id based cryptosystems with pairing on elliptic curve, in *2003 Symposium on Cryptography and Information Security—SCIS'2003*, Hamamatsu, Japan (2003). http://eprint.iacr.org/2003/054

[44] A. Shamir, Identity-based cryptosystems and signature schemes, in *Advances in Cryptology—CRYPTO'84*, Santa Barbara, CA, USA, August 19–23, ed. by G.R. Blakley, D. Chaum. Lecture Notes in Computer Science, vol. 196 (Springer, Berlin, 1985), pp. 47–53

[45] V. Shoup, *A Computational Introduction to Number Theory and Algebra* (Cambridge University Press, Cambridge, 2005)

[46] B.R. Waters, Efficient identity-based encryption without random oracles, in *Advances in Cryptology—EUROCRYPT 2005*, Aarhus, Denmark, May 22–26, ed. by R. Cramer. Lecture Notes in Computer Science, vol. 3494 (Springer, Berlin, 2005), pp. 114–127