Identity-Based Lossy Trapdoor Functions: New Definitions, Hierarchical Extensions, and Implications

Alex Escala[1], Javier Herranz[2], Benoît Libert[3], and Carla Ràfols[4]

[1] Scytl Secure Electronic Voting (Spain)
[2] Universitat Politècnica de Catalunya - BarcelonaTech, Dept. Matemàtica Aplicada IV (Spain)
[3] Technicolor (France)
[4] Ruhr-Universität Bochum, Horst Görtz Institut für IT-Sicherheit (Germany)

Abstract. Lossy trapdoor functions, introduced by Peikert and Waters (STOC'08), have received a lot of attention in the last years, because of their wide range of applications. The notion has been recently extended to the identity-based setting by Bellare *et al.* (Eurocrypt'12). An identity-based trapdoor function (IB-TDF) satisfying the lossy property introduced by Bellare *et al.* can be used to construct other cryptographic primitives in the identity-based setting: encryption schemes with semantic security under chosen-plaintext attacks, deterministic encryption schemes, and hedged encryption schemes that maintain some security when messages are encrypted using randomness of poor quality. However, the constructed primitives can be proved secure only against *selective* adversaries who select the target identity upfront.

Our first contribution is an alternative definition for the lossiness of an identity-based trapdoor function. We prove that an IB-TDF satisfying the new property can be used to construct all the aforementioned primitives, in the identity-based setting, with security against *adaptive* adversaries. We further consider the new definition and its implications in the more general scenario of *hierarchical* identity-based cryptography, which has proved very useful both for practical applications and to establish theoretical relations with other cryptographic primitives (including encryption with chosen-ciphertext security or with forward-security).

As a second contribution, we describe a pairing-based hierarchical IB-TDF satisfying the new definition of lossiness against either selective or, for hierarchies of constant depth, adaptive adversaries. This is also the first example of hierarchical trapdoor functions based on traditional (*i.e.*, non-lattice-related) number theoretic assumptions. As a direct consequence of our two contributions, we obtain a hierarchical identity-based (HIB) encryption scheme with chosen-plaintext security, a HIB deterministic encryption scheme and a HIB hedged encryption scheme, all of them with security against adaptive adversaries.

1 Introduction

1.1 (Identity-Based) Lossy Trapdoor Functions

Lossy trapdoor functions, as introduced by Peikert and Waters in [25], have been proved very powerful in theoretical cryptography and received a lot of attention in the recent years (see, e.g., [17, 21, 10, 22, 30]). Lossy trapdoor functions are function families that can be instantiated in two different modes. In the injective mode, the function is injective and can be inverted using a trapdoor. In lossy mode, the function is (highly) non-injective since its image size is much smaller than the size of the domain. The key point is that lossy instantiations of the function must be indistinguishable from injective instantiations.

In their seminal paper [25], Peikert and Waters showed that lossy trapdoor functions provide black-box constructions of chosen-ciphertext secure (IND-CCA) public-key encryption schemes as well as universal one-way and collision-resistant hash functions. Later on, other applications of lossy trapdoor functions were discovered: they gave rise to deterministic encryption schemes [2] in the standard model [6], public-key hedged encryption schemes maintaining some security in the absence of reliable encryption coins [3] and even public-key encryption with selective-opening security [4] (*i.e.*, which offer certain security guarantees in case of sender corruption).

Recently, Bellare, Kiltz, Peikert and Waters [5] introduced the notion of identity-based (lossy) trapdoor function (IB-TDF), which is the analogue of lossy trapdoor functions in the setting of identity-based cryptography [28]. In the identity-based scenario, users' public keys are directly derived from their identities, whereas secret keys are delivered by a trusted master entity. In this way, the need for digital certificates, which usually bind public keys to users in traditional public-key cryptography, is drastically reduced. Throughout the last decade, several generalizations of identity-based cryptography were put forth, including hierarchical identity-based encryption [18], attribute-based encryption [26, 19] or predicate encryption [8, 23]. In the setting of hierarchical identity-based cryptography, identities are organized in a hierarchical way, so that a user who holds the secret key of an identity id can generate, use and distribute valid secret keys for any identity that is a descendant of id in the hierarchy. Hierarchical identity-based encryption (HIBE) is of great interest due to both practical and theoretical reasons. On the practical side, many organizations and systems that may need (identity-based) cryptographic solutions are organized in a hierarchical way. On the theoretical side, generic constructions [11, 12] are known to transform a weakly secure HIBE scheme (*i.e.*, IND-CPA security against selective adversaries) into (public-key) encryption schemes with strong security properties, like chosen-ciphertext security [12] or forward-security [1, 11], where private keys are updated in such a way that past encryptions remain safe after a key exposure.

Bellare *et al.* [5] proposed instantiations of identity-based lossy trapdoor functions based on bilinear maps and on lattices (as noted in [5], almost all IBE schemes belong to these families). Moreover, they show that their definition of

partial-lossiness for identity-based trapdoor functions leads to the same cryptographic results as lossy trapdoor functions, but in the selective identity-based setting only, where the attacker must choose the target identity upfront in the attack game. Namely, in the case of selective adversaries, IB-TDFs satisfying their definition imply identity-based encryption with semantic security, identity-based deterministic encryption and identity-based hedged encryption. In [5], it was left as an open problem to prove that the same results hold in the case of adaptive adversaries.

1.2 Our Two Main Contributions

New Definition of Partial Lossiness and Its Applications. From a theoretical standpoint, we first define a new security property for hierarchical identity-based trapdoor functions (HIB-TDFs). For the particular (non-hierarchical) case of IB-TDFs, the new security property is different to the property of partial lossiness defined by Bellare *et al.* [5]. We show that a HIB-TDF which satisfies this new property can be used to obtain the same kind of results that are derived from standard lossy trapdoor functions [25]. Namely, they lead to standard encryption schemes, to deterministic encryption schemes for block sources, and to non-adaptive hedged encryption schemes (also for block sources), which are secure in the hierarchical identity-based setting, against adaptive-id adversaries. Since IB-TDFs are a particular case of HIB-TDFs, our results for adaptive adversaries solve the above mentioned open problem in [5]. Interestingly, the pairing-based IB-TDF of Bellare *et al.* [5] can be proved to also satisfy the new security property. See the full version of the paper [16] for more details on this. As a consequence, it provides *adaptive*-id secure deterministic and hedged IBE schemes, and not only selectively secure ones as initially believed.

Construction of a Pairing-Based Hierarchical Trapdoor Function. On the constructive side, we focus on pairing-based hierarchical systems and leave possible constructions based on lattices as an open line for future work. Our intuition, however, is that pairing-based HIB-TDFs seem harder to construct than their lattice-based counterpart. Indeed, no hierarchical trapdoor function is currently known to rely on traditional number theoretic assumptions whereas, in the lattice world, constructions have been known since the results of Cash, Hofheinz, Kiltz and Peikert [13].

Using bilinear maps, we build a HIB-TDF and prove that it satisfies our new definition of partial lossiness under mild assumptions in prime order groups. As an intermediate step, we design a hierarchical predicate encryption (HPE) system [27, 24] with suitable anonymity properties. Perhaps surprisingly, although this scheme is proved secure only against weak *selective* adversaries (who select their target attribute set before seeing the public parameters), we are able to turn it into a HIB-TDF providing security (namely, our new version of partial lossiness) against *adaptive* adversaries for hierarchies of constant depth. To the best of our knowledge, our HIB-TDF gives rise to the first hierarchy of trapdoor functions which does not rely on lattices: realizing such a hierarchy using number

theoretic techniques was identified as an open problem in [13].

Beyond its hierarchical nature, our construction brings out an alternative design principle for (H)IB-TDFs. The idea is to rely on hierarchical predicate encryption (HPE) to deal with hierarchies. Namely, public parameters consist of a matrix of HPE encryptions and, when the function has to be evaluated, the latter matrix is turned into a matrix of (anonymous) HIBE ciphertexts. The homomorphic properties of the underlying HIBE then make it possible to evaluate the function while guaranteeing a sufficient amount of lossiness in lossy mode. It seems possible to abstract away the properties of the underlying HPE system in order to obtain a HIB-TDF via a semi-generic transformation. However, the HPE scheme we describe seems to be the only candidate with the required algebraic structure.

While the pairing-based IB-TDF construction of Bellare *et al.* [5] builds on an adaptively secure anonymous IBE, our HIB-TDF is obtained from a *selective* weakly attribute-hiding HPE system. This result is somewhat incomparable with [5]: on the one hand, we start from a more powerful primitive – because predicate encryption implies anonymous IBE – but, on the other hand, we need a weaker security level to begin with. Both (H)IB-TDF constructions rely on specific algebraic properties in the underlying IBE/HPE and neither is generic.

1.3 Implications

Combining our HIB-TDF with the theoretical implications of our new security property, we obtain: (1) a modular way to build adaptive-id secure HIBE schemes from HIB-LTDFs, (2) the first secure deterministic HIBE scheme for block sources[5], (3) the first HIBE scheme, for block sources, that (non-adaptively) hedges against bad randomness, as advocated by Bellare *et al.* [3]. All these schemes are secure against both selective and adaptive-id adversaries.

In the case of adaptive adversaries, these results only hold for hierarchies of constant depth (said otherwise, we do not provide full security). However, using our definition of partial lossiness or that of Bellare *et al.* [5], this appears very difficult to avoid. The reason is that both definitions seem inherently bound to the partitioning paradigm. Namely, they assume the existence of alternative public parameters, called *lossy parameters*, where the identity space is partitioned into subsets of injective and lossy identities (this is, identities which lead to injective and lossy functions respectively). The definition of [5] intuitively captures that a fraction δ of identities are lossy in the case of lossy parameters. In the hierarchical setting, the analogy with HIBE schemes suggests that all ancestors of a lossy identity be lossy themselves. Hence, unless one can make sure that certain lossy identities only have lossy descendants, the fraction δ seems doomed to exponentially decline with the depth of the hierarchy.

Finally, due to the results of Canetti, Halevi and Katz [11], our construction also implies the first forward-secure deterministic and hedged public-key encryption schemes (note that, as pointed out in [11], selective security suffices to give

[5] See [31] for a recent and independent construction, in the (non-hierarchical) IBE case.

forward-secure cryptosystems). Although our scheme is not practical due to large ciphertexts and keys, it provides the first feasibility results in these directions.

2 Background

2.1 Some Complexity Assumptions

We will consider groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$ of prime order p for which an asymmetric bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ is efficiently computable. We will assume that the DDH assumption holds in both $\mathbb{G}$ and $\hat{\mathbb{G}}$, which implies that no isomorphism is efficiently computable between $\mathbb{G}$ and $\hat{\mathbb{G}}$. The assumptions that we need are sometimes somewhat stronger than DDH. However, they have *constant* size (*i.e.* we de not rely on q-type assumptions) and were previously used in [15].

The Bilinear Diffie Hellman Assumption (BDH): in asymmetric bilinear groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$ of prime order p, no PPT adversary can distinguish the distribution $D_1 = \{(g,\ g^a,\ g^c,\ \hat{g},\ \hat{g}^a,\ \hat{g}^b,\ e(g,\hat{g})^{abc}) \mid a, b, c \xleftarrow{R} \mathbb{Z}_p\}$, from $D_2 = \{(g,\ g^a,\ g^c,\ \hat{g},\ \hat{g}^a,\ \hat{g}^b,\ e(g,\hat{g})^z) \mid a, b, c, z \xleftarrow{R} \mathbb{Z}_p\}$.

The $\mathcal{P}$-BDH$_1$ Assumption: in asymmetric bilinear groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$ of prime order p, the distribution $D_1 = \{(g, g^b, g^{ab}, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, g^{abc}) \mid a, b, c \xleftarrow{R} \mathbb{Z}_p\}$ is indistinguishable from $D_2 = \{(g, g^b, g^{ab}, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, g^z) \mid a, b, c, z \xleftarrow{R} \mathbb{Z}_p\}$ for any PPT algorithm.

The DDH$_2$ Assumption: in asymmetric bilinear groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$ of prime order p, the distribution $D_1 = \{(g,\ \hat{g},\ \hat{g}^a,\ \hat{g}^b,\ \hat{g}^{ab}) \mid a, b \xleftarrow{R} \mathbb{Z}_p\}$ is computationally indistinguishable from $D_2 = \{(g,\ \hat{g},\ \hat{g}^a,\ \hat{g}^b,\ \hat{g}^z) \mid a, b, z \xleftarrow{R} \mathbb{Z}_p\}$.

2.2 Hierarchical Identity-Based (Lossy) Trapdoor Functions

This section recalls formal definitions of (hierarchical) identity-based lossy trapdoor function.

Syntax. A hierarchical identity-based trapdoor function (HIB-TDF) is a tuple of efficient algorithms $\mathsf{HF} = (\mathsf{HF.Setup}, \mathsf{HF.MKg}, \mathsf{HF.Kg}, \mathsf{HF.Del}, \mathsf{HF.Eval}, \mathsf{HF.Inv})$. The setup algorithm $\mathsf{HF.Setup}$ takes as input a security parameter $\varrho \in \mathbb{N}$, the (constant) number of levels in the hierarchy $d \in \mathbb{N}$, the length of the identities $\mu \in \mathsf{poly}(\varrho)$ and the length of the function inputs $n \in \mathsf{poly}(\varrho)$, and outputs a set of global public parameters pms, which specifies an input space InpSp, an identity space IdSp and the necessary mathematical objects and hash functions. The master key generation algorithm $\mathsf{HF.MKg}$ takes as input pms and outputs a master public key mpk and a master secret key msk. The key generation algorithm $\mathsf{HF.Kg}$ takes as input pms, msk and a hierarchical identity $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell) \in \mathsf{IdSp}$, for some $\ell \geq 1$ and outputs a secret key $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}$. The delegation algorithm $\mathsf{HF.Del}$ takes as input pms, mpk, a hierarchical identity $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, a secret key $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}$ for it, and an additional identity $\mathsf{id}_{\ell+1}$; the output is a secret key $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell,\mathsf{id}_{\ell+1})}$ for the hierarchical identity $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1})$ iff $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1}) \in \mathsf{IdSp}$. The evaluation algorithm $\mathsf{HF.Eval}$ takes as input pms,

mpk, an identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ and a value $X \in \mathsf{InpSp}$; the result of the evaluation is denoted as C. Finally, the inversion algorithm $\mathsf{HF.Inv}$ takes as input pms, mpk, a hierarchical identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, a secret key $\mathbf{SK}_{\mathsf{id}}$ for it and an evaluation C, and outputs a value $\tilde{X} \in \mathsf{InpSp}$.

A HIB-TDF satisfies the property of correctness if we have the equaltity $\mathsf{HF.Inv}\big(\mathsf{pms}, \mathsf{mpk}, \mathsf{id}, \mathbf{SK}_{\mathsf{id}}, \mathsf{HF.Eval}\big(\mathsf{pms}, \mathsf{mpk}, \mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), X\big)\big) = X$, for any $X \in \mathsf{InpSp}$, any $\mathsf{pms}, (\mathsf{mpk}, \mathsf{msk})$ generated by $\mathsf{HF.Setup}$ and $\mathsf{HF.MKg}$, any hierarchical identity $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell) \in \mathsf{IdSp}$ and any key $\mathbf{SK}_{(\mathsf{id}_1, \ldots, \mathsf{id}_\ell)}$ generated either by running $\mathsf{HF.Kg}\big(\mathsf{pms}, \mathsf{msk}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)\big)$ or by applying the delegation algorithm $\mathsf{HF.Del}$ to secret keys of shorter hierarchical identities.

Before formalizing the new definition of partial lossiness for a HIB-TDF, let us recall the notion of *lossiness*: if f is a function with domain $\mathrm{Dom}(f)$ and image $\mathrm{Im}(f) = \{f(x) \ : \ x \in \mathrm{Dom}(f)\}$, we say that f is ω-lossy if $\lambda(f) \geq \omega$, where $\lambda(f) = \log \frac{|\mathrm{Dom}(f)|}{|\mathrm{Im}(f)|}$.

To define lossiness for HIB-TDFs, it is useful to consider extended HIB-TDFs, which differ from standard HIB-TDFs in that, in the latter, the algorithm $\mathsf{HF.Setup}$ specifies in pms an auxiliary input space AuxSp, and $\mathsf{HF.MKg}$ admits an auxiliary input $aux \in \mathsf{AuxSp}$. Given a HIB-TDF $\mathsf{HF} = (\mathsf{HF.Setup}, \mathsf{HF.MKg}, \mathsf{HF.Kg}, \mathsf{HF.Del}, \mathsf{HF.Eval}, \mathsf{HF.Inv})$, a *sibling* for HF is an extended HIB-TDF $\mathsf{LHF} = (\mathsf{HF.Setup}, \mathsf{LHF.MKg}, \mathsf{LHF.Kg}, \mathsf{HF.Del}, \mathsf{HF.Eval}, \mathsf{HF.Inv})$ whose delegation, evaluation and inversion algorithms are those of HF, and where an auxiliary space AuxSp is contained in $\mathsf{pms} \leftarrow \mathsf{HF.Setup}(\varrho)$, so that $\mathsf{IdSp} \subset \mathsf{AuxSp}$.

Looking ahead, we will define, as in [5], two different experiments: one corresponding to the standard setup and one corresponding to the lossy setup, in one of them the experiment will interact with a standard HIB-TDF, in the other one with a sibling in which some identities lead to lossy evaluation functions. The notion of extended HIB-TDF will serve to construct both of these functions as an extended HIB-TDF but with different auxiliary inputs $\boldsymbol{y}^{(0)}, \boldsymbol{y}^{(1)}$.

3 A New Security Definition for (H)IB-TDFs

The basic security property of a trapdoor function is *one-wayness*, which means that the function is hard to invert without the suitable secret key. In the identity-based setting, one-wayness is required to hold even when the adversary has access to secret keys for some identities. *Partial lossiness* for identity-based trapdoor functions was introduced in [5], where it was proved to imply one-wayness. Roughly speaking, partial lossiness requires that the weighted difference of the probability that any adversary outputs 1 in the lossy or in the real experiment is negligible. These weights account for the fact that, in the lossy experiment in the adaptive case, some identities may lead to lossy functions, which can be detected by any adversary $\mathcal{A}$ that queries the secret key for such an identity. As a result, there is an asymmetry when comparing the real and the lossy experiments which is compensated by the weights.

For the selective case, the weights can simply be set to 1 and it can be proved that an IB-TDF satisfying their notion of partial lossiness in the selective

scenario can be used to build: (1) identity-based encryption (IBE) schemes with selective IND-CPA security, (2) selectively secure deterministic IBE schemes, (3) selectively secure hedged IBE schemes. However, these results are not known to be true in the adaptive setting. In fact, the definition is not known to imply the IND-ID-CPA security of the resulting IBE scheme in the adaptive-id scenario.

To address this question, we propose an alternative definition for the partial lossiness of (hierarchical) identity-based trapdoor functions — in particular, the definition is also different from the one of Bellare *et al.* when the hierarchy depth is equal to 1, the case considered in [5]. We will show that a HIB-TDF satisfying this new definition gives, in the adaptive-id case, a secure construction of the same primitives we mentioned for the selective-id case.

3.1 The Formal Definition

As in [5], we define two different experiments, a lossy experiment and a real experiment. For any adversary $\mathcal{A}$ against a HIB-TDF, the $\mathsf{REAL}^{\mathcal{A}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}$ experiment and the $\mathsf{LOSSY}^{\mathcal{A}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}$ experiment are parameterized by the security parameter ϱ (which is usually omitted in the notation) and some values $\zeta(\varrho), \omega(\varrho)$. The experiment also takes as input the specification of some algorithm $\mathcal{P}$ which takes as input $\zeta, \mathsf{pms}, \mathsf{mpk}_1, \mathsf{msk}_1, IS, \mathsf{id}^\star$, and outputs a bit d_2. This algorithm must be efficient for any non-negligible ζ. To simplify notations, we write REAL instead of $\mathsf{REAL}^{\mathcal{A}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}$ and LOSSY instead of $\mathsf{LOSSY}^{\mathcal{A}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}$.

We present the two experiments as a single experiment depending on a bit β: the challenger $\mathcal{C}$, who interacts with the adversary $\mathcal{A}$, runs either REAL if $\beta = 0$ or LOSSY if $\beta = 1$. Also, some instructions of both experiments depend on whether *selective* or *adaptive* security is being considered. We say that a hierarchical identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ is a *prefix* of another one $\mathsf{id}^\star = (\mathsf{id}^\star_1, \ldots, \mathsf{id}^\star_{\ell^\star})$ if $\ell \leq \ell^\star$ and $\mathsf{id}_i = \mathsf{id}^\star_i$ for every $i = 1, \ldots, \ell$. We denote this fact by $\mathsf{id} \leq \mathsf{id}^\star$.

0. First, $\mathcal{C}$ chooses global parameters pms by running $\mathsf{HF.Setup}$. The parameters pms are given to the adversary $\mathcal{A}$, who replies by choosing a hierarchical identity $\mathsf{id}^\dagger = (\mathsf{id}^\dagger_1, \ldots, \mathsf{id}^\dagger_{\ell^\dagger})$, for some $\ell^\dagger \leq d$.
1. The challenger $\mathcal{C}$ runs $(\mathsf{mpk}_0, \mathsf{msk}_0) \leftarrow \mathsf{HF.MKg}(\mathsf{pms})$ and $(\mathsf{mpk}_1, \mathsf{msk}_1) \leftarrow \mathsf{LHF.MKg}(\mathsf{pms}, aux = \mathsf{id}^\dagger)$. The adversary $\mathcal{A}$ receives mpk_β and lists $IS \leftarrow \emptyset$, $QS \leftarrow \emptyset$ are initialized.
2. $\mathcal{A}$ can make adaptive queries for hierarchical identities $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$.
   - **Create-key**: $\mathcal{A}$ chooses an identity id and $\mathcal{C}$ creates a private key SK_{id}. If $\beta = 0$, SK_{id} is created by running $\mathsf{HF.Kg}(\mathsf{pms}, \mathsf{msk}_0, \mathsf{id})$. If $\beta = 1$, it is created by running $\mathsf{LHF.Kg}(\mathsf{pms}, \mathsf{msk}_1, \mathsf{id})$. The list QS is updated as $QS = QS \cup \{\mathsf{id}\}$.
   - **Create-delegated-key**: $\mathcal{A}$ provides a tuple $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ and $\mathsf{id}_{\ell+1}$ such that $\mathsf{id} \in QS$. The challenger $\mathcal{C}$ then computes a delegated key $SK_{\mathsf{id}'}$ for $\mathsf{id}' = (\mathsf{id}_1, \ldots, \mathsf{id}_{\ell+1})$ by running the delegation algorithm $\mathsf{HF.Del}\big(\mathsf{pms}, \mathsf{mpk}_\beta, SK_{\mathsf{id}}, \mathsf{id}_{\ell+1}\big)$ before setting $QS = QS \cup \{\mathsf{id}'\}$.
   - **Reveal-key**: $\mathcal{A}$ provides id with the restriction that if $\mathcal{A}$ is selective, then $\mathsf{id} \not\leq \mathsf{id}^\dagger$. $\mathcal{C}$ returns $\perp$ if $\mathsf{id} \notin QS$. Otherwise, SK_{id} is returned to $\mathcal{A}$

and the list IS is updated as $IS = IS \cup \{\mathsf{id}\}$.

3. $\mathcal{A}$ outputs a hierarchical identity $\mathsf{id}^\star = (\mathsf{id}_1^\star, \ldots, \mathsf{id}_{\ell^\star}^\star)$. If $\mathcal{A}$ is selective, then $\mathsf{id}^\star = \mathsf{id}^\dagger$. In the adaptive case, no element of IS can be a prefix of $\mathsf{id}^\star$.
4. The adversary can make adaptive queries such as the ones described in step 2, with the restriction that for **Reveal-key** queries the id provided by $\mathcal{A}$ must satisfy $\mathsf{id} \not\leq \mathsf{id}^\star$.
5. The adversary outputs a bit $d_{\mathcal{A}} \in \{0,1\}$. Let d_1 be the bit $d_1 := \big(\forall\ \mathsf{id} \in IS, \lambda\left(\mathsf{HF.Eval}(\mathsf{pms}, \mathsf{mpk}_1, \mathsf{id}, \cdot)\right) = 0\big) \wedge \big(\lambda\left(\mathsf{HF.Eval}(\mathsf{pms}, \mathsf{mpk}_1, \mathsf{id}^\star, \cdot)\right) \geq \omega\big)$.
6. The challenger $\mathcal{C}$ sets d_2 to be the output of the pre-output stage $\mathcal{P}$ with input $\zeta, \mathsf{pms}, \mathsf{mpk}_1, \mathsf{msk}_1, IS, \mathsf{id}^\star$.
7. The final output of the experiment consists of the bits $\{d_{\mathcal{A}},\ d^{\mathcal{A}}_{\neg abort}\}$, where $d^{\mathcal{A}}_{\neg abort} = d_1 \wedge d_2 \in \{0,1\}$.

For notational convenience, from now on, let us define $d^{\mathcal{A}}_{exp} = d_{\mathcal{A}} \wedge d^{\mathcal{A}}_{\neg abort}$.

Definition 1. *A HIB-TDF is (ω, δ)-partially lossy if it admits a sibling and an efficient pre-output stage $\mathcal{P}$ such that for all PPT adversaries $\mathcal{A}$ and for all non-negligible ζ, there exist two non-negligible values ϵ_1, ϵ_2 such that $\delta = \epsilon_1\epsilon_2$ is non-negligible and the following three conditions hold:*

(i) the following advantage function is negligible in the security parameter ϱ:

$$\mathbf{Adv}^{\text{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A}) = |\Pr[d^{\mathcal{A}}_{exp} = 1|\ \mathsf{REAL}]\ -\ \Pr[d^{\mathcal{A}}_{exp} = 1|\ \mathsf{LOSSY}]| \quad (1)$$

(ii) $\Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{REAL}] \geq \epsilon_1$.

(iii) if $\mathcal{A}$ is such that $\Pr[d_{\mathcal{A}} = 1 \mid \mathsf{REAL}]\ -\ \frac{1}{2} > \zeta$, *then*

$$\Pr[d_{\mathcal{A}} = 1 \mid \mathsf{REAL} \wedge d^{\mathcal{A}}_{\neg abort} = 1] - \frac{1}{2} > \epsilon_2 \cdot \zeta, \quad (2)$$

where δ may be a function of q the maximal number of key queries of $\mathcal{A}$.

Some Intuition. As we mentioned, to account for the asymmetry between the real and the lossy experiment, Bellare *et al.* defined the advantage of a distinguisher among the lossy and real experiments as the *weighted* difference of the probability of outputting 1 in the real case minus the same probability in the lossy case. Our solution is different. We always execute in parallel two instances of the master key generation protocol, one in the real and one in the lossy mode (the adversary does not notice this). The experiments output a bit d_1 which is computed in the same way in both the real and lossy settings and which depends on the secret key queries and the challenge identity chosen by the adversary: for example, if a query would force the LOSSY experiment to output $d_1 = 0$ indicating that the adversary queried for a secret key of a lossy identity, then it also forces the REAL experiment to output $d_1 = 0$. For the sake of intuition, let us think of $d^{\mathcal{A}}_{\neg abort}$ as the bit d_1 and temporarily ignore d_2, whose purpose is explained later. By defining the output of the experiment as the logical AND

between the output $d_{\mathcal{A}}$ of the adversary and a bit $d^{\mathcal{A}}_{\neg abort}$, we just avoid having to introduce weights in condition (i) in Definition 1. This is a difference with [5] which is crucial to prove that lossy identity-based trapdoor functions imply other primitives in the adaptive-id case.

Condition (i) can be seen as the natural (non-weighted) analogue of the security definition of [5], while the other conditions might look more artificial. We provide some more intuition on why condition (i) alone is not useful to guarantee that security reductions can be done from a scheme Π built from an HIB-TDF to the HIB-TDF itself in the adaptive setting. First, we add condition (ii) to rule out some cases in which the condition (i) would be trivial to satisfy, like the case where the procedure $\mathcal{P}$ aborts with overwhelming probability or the case where the sibling admits only lossy identities: in any of these scenarios, we would have $d^{\mathcal{A}}_{\neg abort} = 1$ with negligible probability, which would render the scheme useless.

A more serious problem, which motivates condition (iii), is that, in the reduction, the output of a potential adversary $\mathcal{A}$ with meaningful advantage ε against Π does not necessarily help to contradict condition (i) because the output of $\mathcal{A}$ needs to be conditioned to $d^{\mathcal{A}}_{\neg abort} = 1$ and the output of $\mathcal{A}$ may not be independent of $d^{\mathcal{A}}_{\neg abort} = 1$. To solve that, the reduction might try to abort after certain events, for example after some secret key queries have been done. However, such events could be related to the underlying HIB-TDF scheme, so the reduction would not be black-box. Condition (iii) guarantees that, if an adversary has some meaningful advantage against the scheme built from the HIB-TDF, it will also have meaningful advantage when $d_{\neg abort} = 1$, this is, when all the secret key queries correspond to injective identities and the challenge identity is lossy. Roughly said, this condition ensures that the probability of aborting is somewhat independent of the behavior of any computationally bounded adversary.

We have not discussed the role of d_2 yet. If $d_{\neg abort}$ was just defined as d_1, condition (iii) would be quite hard to satisfy: intuitively we would not be allowing the security reduction to make extra aborts related to events which depend on the HIB-TDF, which is unnecessarily restrictive. To handle this problem, we allow the experiment to consider an efficient algorithm $\mathcal{P}$, which depends on the HIB-TDF and outputs a bit d_2, and we define $d_{\neg abort} = d_1 \wedge d_2$. Finally, we stress that the incorporation of algorithm $\mathcal{P}$ results in a more general and flexible security definition: although one could define the security of HIB-TDF without taking into account the existence of such an algorithm $\mathcal{P}$, it would make it more difficult for a HIB-TDF to satisfy it. On the other hand, if $\mathcal{P}$ is the trivial algorithm which always outputs $d_2 = 1$, this is equivalent to considering the security definition without the algorithm $\mathcal{P}$, which is enough to prove the security of our HIB-TDF against selective adversaries, indeed. To prove the security of our HIB-TDF against adaptive adversaries, we will define $\mathcal{P}$ as the artificial abort stage in the security proof of Waters' IBE scheme [29].

3.2 Implications of Lossy (H)IB-TDFs: the Example of (H)IBE

Using the same argument as in [5], it is quite easy to prove that a HIB-TDF which enjoys the new version of the partial lossiness property is already one-way, in both the selective and adaptive settings. In this section we prove that a HIB-TDF which satisfies our new security definition can be used to build other primitives in the hierarchical identity-based scenario, with security against adaptive adversaries. We detail the example of hierarchical identity-based encryption (HIBE) with IND-CPA security[6]. The construction is the direct adaptation of the Peikert-Waters construction [25] in the public-key setting.

Let HF be a HIB-TDF with message space $\{0,1\}^n$ and (ω, δ) partial lossiness, and $\mathcal{H}$ a family of pairwise independent hash functions from $\{0,1\}^n$ to $\{0,1\}^l$ where $l \leq \omega - 2\lg(1/\epsilon_{LHL})$ for some negligible ϵ_{LHL}. The HIBE scheme has message space $\{0,1\}^l$. Its setup, key generation and key delegation algorithms are basically the same ones as those for HF, the rest are as follows:

$\mathbf{MKGen}(\mathsf{pms})$	$\mathbf{Enc}(\mathsf{pms}, \mathsf{mpk}, m, \mathsf{id})$	$\mathbf{Dec}(\mathsf{pms}, \mathsf{mpk}, \mathbf{SK}_{\mathsf{id}}, C, \mathsf{id})$
$(\mathsf{mpk}', \mathsf{msk}) \leftarrow$	$x \leftarrow \{0,1\}^n$	$x = \mathsf{HF.Inv}(\mathsf{pms},$
$\quad \mathsf{HF.MKg}(1^k)$	$c_1 = \mathsf{HF.Eval}(\mathsf{pms}, \mathsf{mpk}', \mathsf{id}, x)$	$\quad \mathsf{mpk}', \mathbf{SK}_{\mathsf{id}}, c_1, \mathsf{id})$
$h \leftarrow \mathcal{H}$	$c_2 = h(x) \oplus m$	$m = c_2 \oplus h(x)$
$\mathsf{mpk} = (\mathsf{mpk}', h)$	Return $C = (c_1, c_2)$	Return m
Return mpk		

We prove the following theorem.

Theorem 1. *If HF is (ω, δ)-partially lossy for some non-negligible value of δ, then the HIBE scheme Π described is IND-ID-CPA secure. In particular, for every IND-ID-CPA adversary $\mathcal{B}$ against Π there exists a PPT adversary $\mathcal{A}$ against HF such that $\mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A}) \geq \frac{2}{3} \cdot \delta \cdot \mathbf{Adv}^{\mathrm{ind-id-cpa}}(\mathcal{B}) - \nu(\varrho)$, for some negligible function ν. Both adversaries $\mathcal{A}$ and $\mathcal{B}$ run in comparable times; whenever $\mathcal{B}$ is selective, so is $\mathcal{A}$ (for their respective experiments)*

Proof. Let us assume that an adversary $\mathcal{B}$ has advantage at least ζ in breaking the IND-ID-CPA security of the HIBE scheme Π, for some non-negligible ζ. We build an adversary $\mathcal{A}$ that breaks the condition (i) of Definition 1 assuming that conditions (ii) and (iii) are satisfied. Our adversary $\mathcal{A}$, who interacts with a challenger that runs either the experiment REAL or the experiment LOSSY, proceeds to simulate the challenger in the IND-ID-CPA game with $\mathcal{B}$ as follows.

Our adversary$\mathcal{A}$ forwards an identity $\mathsf{id}^\dagger$ to its challenger, which is an arbitrary identity in the adaptive case or corresponds to the challenge identity chosen by $\mathcal{B}$ in the selective case. When the challenger runs the setup and gives the output to $\mathcal{A}$, $\mathcal{A}$ forwards this information to $\mathcal{B}$ together with a hash function $h \leftarrow \mathcal{H}$. When $\mathcal{B}$ asks for a secret key for a hierarchical identity id, $\mathcal{A}$ forwards the query to the experiment and relays the latter's reply to $\mathcal{B}$. At some point, $\mathcal{B}$ outputs $(m_0, m_1, \mathsf{id}^\star)$, with $\mathsf{id}^\dagger = \mathsf{id}^\star$ in the selective case. Adversary $\mathcal{A}$ then

[6] The cases of deterministic HIBE and hedged HIBE are discussed in the full version of this paper [16].

forwards $\mathsf{id}^\star$ to its challenger, chooses $\gamma \leftarrow \{0,1\}$ at random and encrypts m_γ under the identity $\mathsf{id}^\star$. After some more secret key queries, $\mathcal{B}$ outputs a guess γ' and $\mathcal{A}$ outputs $d_{\mathcal{A}} = 1$ if $\gamma = \gamma'$ and $d_{\mathcal{A}} = 0$ otherwise.

In the REAL setting, we have $\Pr[\gamma' = \gamma |\ \mathsf{REAL}] - \frac{1}{2} = \Pr[d_{\mathcal{A}} = 1 |\ \mathsf{REAL}] - \frac{1}{2} \geq \zeta$, since $\mathcal{A}$ perfectly simulated the IND-ID-CPA game with $\mathcal{B}$. This inequality can be combined with conditions (ii) and (iii) of the definition of (ω, δ)-partial lossiness (which we assume to be satisfied by HF), and we obtain

$$\Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{REAL}] \cdot \left(\Pr[d_{\mathcal{A}} = 1 \mid \mathsf{REAL} \wedge d^{\mathcal{A}}_{\neg abort} = 1] - \frac{1}{2} \right) > \epsilon_1 \epsilon_2 \zeta. \quad (3)$$

On the other hand, in the LOSSY setting when $\mathsf{id}^\star$ is lossy, the advantage of $\mathcal{B}$ in guessing γ is negligible. Indeed, since h is a pairwise independent hash function, the Leftover Hash Lemma [20] (more precisely, its variant proved in [14]) implies that the distribution of c_2 given c_1 is statistically uniform. We thus have $\Pr[d_{\mathcal{A}} = 1 |\ \mathsf{LOSSY} \wedge d^{\mathcal{A}}_{\neg abort} = 1] \leq 1/2 + \epsilon_{LHL}$, for some negligible function ϵ_{LHL}. Since $d^{\mathcal{A}}_{exp} = d^{\mathcal{A}}_{\neg abort} \wedge d_{\mathcal{A}}$, we can express $\Pr[d^{\mathcal{A}}_{exp} = 1 \mid \mathsf{LOSSY}]$ as

$$\begin{aligned}
&\Pr[d_{\mathcal{A}} = 1 \mid \mathsf{LOSSY} \wedge d^{\mathcal{A}}_{\neg abort} = 1] \Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{LOSSY}] \\
&\quad \leq \big(\frac{1}{2} + \epsilon_{LHL}\big) \cdot \Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{LOSSY}] \\
&\quad \leq \frac{1}{2} \cdot \big(\Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{REAL}] + \mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A}) \big) + \nu, \qquad (4)
\end{aligned}$$

for some negligible function $\nu \in \mathsf{negl}(\varrho)$. The last equality follows from the fact that $\Pr[d^{\mathcal{A}}_{\neg abort} = 1 |\ \mathsf{LOSSY}] - \Pr[d^{\mathcal{A}}_{\neg abort} = 1 |\ \mathsf{REAL}] \leq \mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A})$: otherwise, we can build a distinguisher[7] against condition (i) of the partial lossiness definition. If we plug (4) into the definition of $\mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A})$, we find

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A}) &= \big|\Pr[d^{\mathcal{A}}_{exp} = 1 \mid \mathsf{REAL}] - \Pr[d^{\mathcal{A}}_{exp} = 1 \mid \mathsf{LOSSY}]\big| \\
&\geq \left| \Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{REAL}] \cdot \left(\Pr[d_{\mathcal{A}} = 1 \mid \mathsf{REAL} \wedge d^{\mathcal{A}}_{\neg abort} = 1] - \frac{1}{2} \right) \right| \\
&\quad - \frac{1}{2} \cdot \mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A}) - \nu,
\end{aligned}$$

so that there exists $\tilde{\nu} \in \mathsf{negl}(\varrho)$ such that $\mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A})$ is at least

$$\frac{2}{3} \cdot \left| \Pr[d^{\mathcal{A}}_{\neg abort} = 1 \mid \mathsf{REAL}] \cdot \left(\Pr[d_{\mathcal{A}} = 1 \mid \mathsf{REAL} \wedge d^{\mathcal{A}}_{\neg abort} = 1] - \frac{1}{2} \right) \right| - \tilde{\nu}.$$

Using (3) and $\delta = \epsilon_1 \epsilon_2$, we have that the right-hand-side member of the above expression is at least $(2/3) \cdot \delta \cdot \zeta - \nu$. This means that $\mathbf{Adv}^{\mathrm{lossy}}_{\mathsf{HF},\mathsf{LHF},\mathcal{P},\omega,\zeta}(\mathcal{A})$ is non-negligible, which contradicts condition (i). □

[7] This distinguisher $\mathcal{A}_1$ is obtained from $\mathcal{A}$ by ignoring $d_{\mathcal{A}} \in \{0,1\}$ and replacing it by a 1, so that $d^{\mathcal{A}}_{\neg abort} = d^{\mathcal{A}}_{exp}$.

4 A Hierarchical Identity-Based (Lossy) Trapdoor Function

The design of our new HIB-TDF and its security analysis use as a key ingredient an HPE scheme which is described in the full version of the paper [16]. Let us first provide some intuition on the reason why a HPE scheme simplifies our task.

The pairing-based IB-TDF of Bellare *et al.* [5] uses an anonymous IBE scheme as a building block. To construct a HIB-TDF, a natural idea is thus to use an anonymous HIBE system. One difficulty is that, at least in the world of pairings, anonymous IBE schemes are usually harder to extend to a hierarchy than non-anonymous ones. Indeed, private keys have to contain extra randomization components because, if the randomization material were included in the public parameters, ciphertexts would betray the identity of receivers. Moreover, as already mentioned in [5], anonymity is not sufficient by itself: what we need is a way to properly embed an auxiliary input in the public parameters without the adversary noticing the difference between two distinct auxiliary inputs. Adapting the Boyen-Waters anonymous HIBE [9] to achieve this is not straightforward. On the other hand, HPE schemes make it possible to naturally embed auxiliary inputs in the attribute vectors of HPE ciphertexts, which are included in the public parameters of the function. When the function has to be evaluated for a specific identity, our construction uses a mechanism to turn HPE ciphertexts into a matrix of HIBE ciphertexts and this is where the interaction between auxiliary inputs and hierarchical identities leads to functions that can be injective or lossy. From the resulting matrix of HIBE ciphertexts, the function evaluation proceeds by computing a matrix-vector product in the exponent, as done in many lossy TDF construcitons (see, e.g., [25, 17, 21, 30]), and takes advantage of homomorphic properties in the underlying HIBE system.

More precisely, our lossy function is obtained by including a $n \times n$ matrix of HPE ciphertexts in the master public parameters. As in the DDH-based function of [25], each row of the matrix is associated with an encryption exponent, which is re-used throughout the entire row. Each column corresponds to a different set of public parameters in the HPE system.

The HIB-TDF that we construct is actually an extended HIB-TDF, and so the master key generation protocol takes an auxiliary input. Depending on the value of this auxiliary input, we obtain the trapdoor (injective) function or a partially lossy function, used in the security proofs. Actually, all HPE ciphertexts in the above-mentioned matrix correspond to different hierarchical vectors $(\mathbf{y}_1, \ldots, \mathbf{y}_d) \in \mathbb{Z}_p^{d \cdot \mu}$, depending on the auxiliary input. The selective weak attribute-hiding property of the HPE scheme guarantees that the two setups are computationally indistinguishable.

In order to evaluate a function for some hierarchical identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, the first step of the evaluation algorithm computes a transformation on HPE ciphertexts so as to obtain a matrix of Boneh-Boyen HIBE ciphertexts [7] in their anonymized variant suggested by Ducas [15]. During this transformation, a set of inner products $\{\langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1} \rangle\}_{i_1=1}^{\ell}$ is calculated in the exponent in the diagonal entries of the matrix. The transformation provides a $n \times n$ matrix (7) of anony-

mous HIBE ciphertexts that are always well-formed in non-diagonal entries. As for diagonal entries, they contain "perturbed" HIBE ciphertexts: at each level, one ciphertext component contains a perturbation factor of the form $\langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1} \rangle$. In this matrix of HIBE ciphertexts, random encryption exponents are again reused in all positions at each row.

The function evaluation is carried out as in [25], by computing a matrix-vector product in the exponent and taking advantage of homomorphic properties of the HIBE scheme over the randomness space. The function output can be seen as a set of n anonymous HIBE ciphertexts – one for each input bit – which are well-formed ciphertexts if and only if the corresponding input bit is 0 (*i.e.*, if and only if the perturbation factors $\{\langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1} \rangle\}_{i_1=1}^{\ell}$ are left out when computing the matrix-vector product in the exponent). The function is thus inverted by testing the well-formedness of each HIBE ciphertext using the private key.

4.1 Description

HF.Setup(ϱ, d, n, μ)**:** given a security parameter $\varrho \in \mathbb{N}$, the (constant) desired number of levels in the hierarchy $d \in \mathbb{N}$ and integers $\mu, n \in \mathsf{poly}(\varrho)$ specifying the length of identities and that of function inputs, respectively, choose asymmetric bilinear groups $(\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T)$ of prime order $p > 2^{\varrho}$. Define $\mathsf{InpSp} = \{0,1\}^n$, $\Sigma_{\mathsf{ID}} = \{(1, \mathbf{x}) : \mathbf{x} \in \mathbb{Z}_p^{\mu-1}\}$, $\mathsf{IdSp} = \Sigma_{\mathsf{ID}}^{(\leq d)}$ and $\mathsf{AuxSp} = \mathbb{Z}_p^{d \cdot \mu}$. The public parameters are $\mathsf{pms} = \big(p, (\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T), d, n, \mu, \mathsf{InpSp}, \mathsf{IdSp}, \mathsf{AuxSp}\big)$.

Since HF is an extended HIB-TDF, the master key generation algorithm of our HIB-TDF receives an auxiliary input $\mathbf{y} \in \mathsf{AuxSp}$. Here, it is seen as a concatenation of d row vectors $\mathbf{y}_1, \ldots, \mathbf{y}_d \in \mathbb{Z}_p^{\mu}$. Notation $\Delta(i,j)$ is used for the Kronecker's delta function (that is, $\Delta(i,j) = 1$ if $i = j$, and is equal to 0 otherwise).

HF.MKg$(\mathsf{pms}, \mathbf{y})$**:** parse the auxiliary input as $\mathbf{y} = [\mathbf{y}_1 | \ldots | \mathbf{y}_d] \in \mathbb{Z}_p^{d \cdot \mu}$, and proceed as follows.

1. Choose $\alpha_v \xleftarrow{R} \mathbb{Z}_p^*$, $\boldsymbol{\alpha}_w \xleftarrow{R} (\mathbb{Z}_p^*)^n$, and $\boldsymbol{\alpha}_h \xleftarrow{R} (\mathbb{Z}_p^*)^{d \times (\mu+1) \times n}$. Define $v = g^{\alpha_v}$, $\hat{v} = \hat{g}^{\alpha_v}$, $\mathbf{w} = g^{\boldsymbol{\alpha}_w} \in \mathbb{G}^n$ and $\hat{\mathbf{w}} = \hat{g}^{\boldsymbol{\alpha}_w} \in \hat{\mathbb{G}}^n$. Likewise, set up vectors $\mathbf{h} = g^{\boldsymbol{\alpha}_h} \in \mathbb{G}^{d \times (\mu+1) \times n}$ and $\hat{\mathbf{h}} = \hat{g}^{\boldsymbol{\alpha}_h} \in \hat{\mathbb{G}}^{d \times (\mu+1) \times n}$. Define

$$\mathsf{PP}_{core} := \Big(v,\ \{\mathbf{w}[l_1]\}_{l_1=1}^{n},\ \{\mathbf{h}[i_1, i_2, l_1]\}_{i_1 \in \{1,\ldots,d\}, i_2 \in \{0,\ldots,\mu\},\ l_1 \in \{1,\ldots,n\}} \Big)$$

2. For $i_1 = 1$ to d, parse $\mathbf{y}_{i_1}$ as $(\mathbf{y}_{i_1}[1], \ldots, \mathbf{y}_{i_1}[\mu]) \in \mathbb{Z}_p^{\mu}$. For $l_2 = 1$ to n, do the following.
   a. Choose $\mathbf{s}[l_2] \xleftarrow{R} \mathbb{Z}_p^*$ and compute $\mathbf{J}[l_2] = v^{\mathbf{s}[l_2]}$ as well as

$$\mathbf{C}_w[l_2, l_1] = \mathbf{w}[l_1]^{\mathbf{s}[l_2]},$$
$$\mathbf{C}[i_1, i_2, l_2, l_1] = \big(\mathbf{h}[i_1, 0, l_1]^{\mathbf{y}_{i_1}[i_2] \cdot \Delta(l_2, l_1)} \cdot \mathbf{h}[i_1, i_2, l_1]\big)^{\mathbf{s}[l_2]}$$

   for each $i_1 \in \{1, \ldots, d\}$, $i_2 \in \{1, \ldots, \mu\}$, $l_1 \in \{1, \ldots, n\}$.

b. Define a $n \times n$ matrix $\{\mathbf{CT}[l_2, l_1]\}_{l_2, l_1 \in \{1,\ldots,n\}}$ of HPE ciphertexts

$$\mathbf{CT}[l_2, l_1] = \big(\mathbf{J}[l_2], \mathbf{C}_w[l_2, l_1], \{\mathbf{C}[i_1, i_2, l_2, l_1]\}_{i_1 \in \{1,\ldots,d\},\ i_2 \in \{1,\ldots,\mu\}}\big). \quad (5)$$

The master public key consists of $\mathsf{mpk} := \big(\mathsf{PP}_{core}, \{\mathbf{CT}[l_2, l_1]\}_{l_2, l_1 \in \{1,\ldots,n\}}\big)$ while the master secret key is $\mathsf{msk} := \big(\hat{v}, \hat{\mathbf{w}}, \hat{\mathbf{h}}\big)$. For each $l_1 \in \{1, \ldots, n\}$, it will be convenient to view $(\mathsf{PP}_{core}, \mathsf{msk})$ as a vector of HPE master key pairs $(\mathsf{mpk}[l_1], \mathsf{msk}[l_1])$, with

$$\mathsf{mpk}[l_1] = \big(v, \mathbf{w}[l_1], \{\mathbf{h}[i_1, i_2, l_1]\}_{i_1 \in \{1,\ldots,d\}, i_2 \in \{0,\ldots,\mu\}}\big)$$
$$\mathsf{msk}[l_1] = \big(\hat{v}, \hat{\mathbf{w}}[l_1], \{\hat{\mathbf{h}}[i_1, i_2, l_1]\}_{i_1 \in \{1,\ldots,d\}, i_2 \in \{0,\ldots,\mu\}}\big).$$

HF.Kg$\big(\mathsf{pms}, \mathsf{msk}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)\big)$**:** to generate a key for an identity $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell) \in \mathsf{IdSp}$, parse msk as $\big(\hat{v}, \hat{\mathbf{w}}, \hat{\mathbf{h}}\big)$ and id_{i_1} as $\mathsf{id}_{i_1}[1] \ldots \mathsf{id}_{i_1}[\mu]$ for $i_1 = 1$ to ℓ. Choose $\mathbf{r}_w, \mathbf{r}_1, \ldots, \mathbf{r}_\ell \xleftarrow{R} (\mathbb{Z}_p^*)^n$, choose $\mathbf{s} \xleftarrow{R} (\mathbb{Z}_p^*)^{d \times \mu \times \ell \times n}$, $\mathbf{s}' \xleftarrow{R} (\mathbb{Z}_p^*)^{d \times n}$, and $\mathbf{s}_w \xleftarrow{R} (\mathbb{Z}_p^*)^{d \times \mu \times n}$. For each $l_1 \in \{1, \ldots, n\}$, compute the decryption component $\mathbf{SK}_D = (\mathbf{D}, \mathbf{D}_w, \{\mathbf{D}_{i_1}\}_{i_1=1}^{\ell})$ of the key as $\mathbf{D}_{i_1}[l_1] = \hat{v}^{\mathbf{r}_{i_1}[l_1]}$ and

$$\mathbf{D}[l_1] = \prod_{i_1=1}^{\ell} \Big(\prod_{i_2=1}^{\mu} \hat{\mathbf{h}}[i_1, i_2, l_1]^{\mathsf{id}_{i_1}[i_2]} \Big)^{\mathbf{r}_{i_1}[l_1]} \cdot \hat{\mathbf{w}}[l_1]^{\mathbf{r}_w[l_1]},\ \mathbf{D}_w[l_1] = \hat{v}^{\mathbf{r}_w[l_1]}, \quad (6)$$

while the delegation component $\mathbf{SK}_{DL}$ consists of

$$\big(\{\mathbf{K}[j, k, l_1]\}_{j,k,l_1},\ \{\mathbf{L}[j, l_1]\}_{j,l_1},\ \{\mathbf{L}[j, k, i_1, l_1]\}_{j,k,i_1,l_1},\ \{\mathbf{L}_w[j, k, l_1]\}_{j,k,l_1}\big),$$

with $j \in \{\ell+1, \ldots, d\}$, $k \in \{1, \ldots, \mu\}$, $i_1 \in \{1, \ldots, \ell\}$ and $l_1 \in \{1, \ldots, n\}$, as

$$\mathbf{K}[j, k, l_1] = \prod_{i_1=1}^{\ell} \Big(\prod_{i_2=1}^{\mu} \hat{\mathbf{h}}[i_1, i_2, l_1]^{\mathsf{id}_{i_1}[i_2]} \Big)^{\mathbf{s}[j,k,i_1,l_1]} \cdot \hat{\mathbf{h}}[j, k, l_1]^{\mathbf{s}'[j,l_1]} \cdot \hat{\mathbf{w}}[l_1]^{\mathbf{s}_w[j,k,l_1]},$$

$$\mathbf{L}[j, l_1] = \hat{v}^{\mathbf{s}'[j,l_1]}, \qquad \mathbf{L}[j, k, i_1, l_1] = \hat{v}^{\mathbf{s}[j,k,i_1,l_1]} \quad \text{and } \mathbf{L}_w[j, k, l_1] = \hat{v}^{\mathbf{s}_w[j,k,l_1]}.$$

Output $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)} = \big(\mathbf{SK}_D, \mathbf{SK}_{DL}\big)$.

HF.Del$\big(\mathsf{pms}, \mathsf{mpk}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}, \mathsf{id}_{\ell+1}\big)$**:** parse $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}$ as a HF private key of the form $(\mathbf{SK}_D, \mathbf{SK}_{DL})$, and the identifier $\mathsf{id}_{\ell+1}$ as a string $\mathsf{id}_{\ell+1}[1] \ldots \mathsf{id}_{\ell+1}[\mu] \in \Sigma_{\mathsf{ID}}$. The idea is to run, for $l_1 = 1$ to n, the key derivation algorithm $\mathbf{Delegate}(\mathsf{mpk}[l_1], (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}[l_1], \mathsf{id}_{\ell+1})$ of the HPE scheme, as specified in the full version of the paper, where $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}[l_1] = (\mathbf{SK}_D[l_1], \mathbf{SK}_{DL}[l_1])$ is defined by

$$\mathbf{SK}_D[l_1] = (\mathbf{D}[l_1], \mathbf{D}_w[l_1], \{\mathbf{D}_{i_1}[l_1]\}_{i_1=1}^{\ell})$$
$$\mathbf{SK}_{DL}[l_1] = \big(\{\mathbf{K}[j, k, l_1]\}_{j,k},\ \{\mathbf{L}[j, l_1]\}_{j},\ \{\mathbf{L}[j, k, i_1, l_1]\}_{j,k,i_1}, \{\mathbf{L}_w[j, k, l_1]\}_{j,k}\big).$$

Specifically, for $l_1 = 1$ to n, do the following.

1. Randomize $\mathbf{SK}_{DL}[l_1]$ by raising all its component to some $z \xleftarrow{R} \mathbb{Z}_p^*$. Call this new key $\widehat{\mathbf{SK}}_{DL}[l_1]$ and write its elements with a hat (e.g., $\widehat{\mathbf{K}}[j,k,l_1] = \mathbf{K}[j,k,l_1]^z$).
2. Compute a *partial decryption key*

$$\widetilde{\mathbf{K}}[\ell+1,l_1] = \prod_{k=1}^{\mu} \widehat{\mathbf{K}}[\ell+1,k,l_1]^{\mathsf{id}_{\ell+1}[k]} = \prod_{i_1=1}^{\ell} \Big(\prod_{i_2=1}^{\mu} \hat{\mathbf{h}}[i_1,i_2,l_1]^{\mathsf{id}_{i_1}[i_2]} \Big)^{\tilde{\mathbf{s}}[\ell+1,i_1,l_1]} \cdot \Big(\prod_{k=1}^{\mu} \hat{\mathbf{h}}[\ell+1,k,l_1]^{\mathsf{id}_{\ell+1}[k]} \Big)^{\mathbf{s}'[\ell+1,l_1]} \cdot \hat{\mathbf{w}}[l_1]^{\tilde{\mathbf{s}}_w[\ell+1,l_1]},$$

$$\widetilde{\mathbf{L}}[\ell+1,\ell+1,l_1] = \widehat{\mathbf{L}}[\ell+1,l_1],$$

$$\widetilde{\mathbf{L}}[\ell+1,i_1,l_1] = \prod_{k=1}^{\mu} \widehat{\mathbf{L}}[\ell+1,k,i_1,l_1]^{\mathsf{id}_{\ell+1}[k]} = \hat{v}^{\tilde{\mathbf{s}}[\ell+1,i_1,l_1]} \quad \text{for} \quad i_1 \in \{1,\ldots,\ell\},$$

$$\widetilde{\mathbf{L}}_w[\ell+1,l_1] = \prod_{k=1}^{\mu} \widehat{L}_w[\ell+1,k,l_1]^{\mathsf{id}_{\ell+1}[k]} = \hat{v}^{\tilde{\mathbf{s}}_w[\ell+1,l_1]}$$

where we define $\tilde{\mathbf{s}}[\ell+1,i_1,l_1] = z \cdot (\sum_{k=1}^{\mu} \mathbf{s}[\ell+1,k,i_1,l_1] \cdot \mathsf{id}_{\ell+1}[k])$, for $i_1 \in \{1,\ldots,\ell\}$, and $\tilde{\mathbf{s}}_w[\ell+1,l_1] = z \cdot (\sum_{k=1}^{\mu} \mathbf{s}[\ell+1,k,l_1] \cdot \mathsf{id}_{\ell+1}[k])$.
3. For all $j \in \{\ell+2,\ldots,d\}$, $k \in \{1,\ldots,\mu\}$, compute re-randomized versions of the partial decryption key by raising the partial decryption key to random powers $\tau_{j,k} \xleftarrow{R} \mathbb{Z}_p^*$.

$$\mathbf{K}[\ell+1,l_1]^{(j,k)} = \widetilde{\mathbf{K}}[\ell+1,l_1]^{\tau_{j,k}}, \qquad \mathbf{L}_w[\ell+1,l_1]^{(j,k)} = \widetilde{\mathbf{L}}_w[\ell+1,l_1]^{\tau_{j,k}},$$

$$\{\mathbf{L}[\ell+1,i_1,l_1]^{(j,k)} = \widetilde{\mathbf{L}}[\ell+1,i_1,l_1]^{\tau_{j,k}}\}_{i_1=1}^{\ell+1}.$$

These values will be used to compute the delegation component of the new key at step 5.
4. Compute a decryption component $\mathbf{SK}'_D[l_1] = (\mathbf{D}'[l_1], \mathbf{D}'_w[l_1], \{\mathbf{D}'_{i_1}[l_1]\}_{i_1=1}^{\ell+1})$ for the delegated key by setting $\mathbf{D}'[l_1] = \mathbf{D}[l_1] \cdot \widetilde{\mathbf{K}}[\ell+1,l_1]$ as well as $\mathbf{D}'_w[l_1] = \mathbf{D}_w[l_1] \cdot \widetilde{\mathbf{L}}_w[\ell+1,l_1]$. Then, define $\mathbf{D}'_{\ell+1}[l_1] = \widetilde{\mathbf{L}}[\ell+1,\ell+1,l_1]$ and, for each $i_1 \in \{1,\ldots,\ell\}$, set $\mathbf{D}'_{i_1}[l_1] = \mathbf{D}_{i_1}[l_1] \cdot \widetilde{\mathbf{L}}[\ell+1,i_1,l_1]$.
5. Finally, compute a delegation component for the delegated key. For each $j \in \{\ell+2,\ldots,d\}$, set $\mathbf{L}'[j,l_1] = \widehat{\mathbf{L}}[j,l_1]$. Then, for each $k \in \{1,\ldots,\mu\}$, $i_1 \in \{1,\ldots,\ell+1\}$, set $\mathbf{K}'[j,k,l_1] = \widehat{\mathbf{K}}[j,k,l_1] \cdot \mathbf{K}[\ell+1,l_1]^{(j,k)}$ and

$$\mathbf{L}'_w[j,k,l_1] = \widehat{\mathbf{L}}_w[j,k,l_1] \cdot \mathbf{L}_w[\ell+1,l_1]^{(j,k)}$$
$$\mathbf{L}'[j,k,i_1,l_1] = \widehat{\mathbf{L}}[j,k,i_1,l_1] \cdot \mathbf{L}[\ell+1,i_1,l_1]^{(j,k)},$$

with $\widehat{\mathbf{L}}[j,k,\ell+1,l_1] = 1$ for all j,k. The delegation component $\mathbf{SK}'_{DL}$ is

$$\mathbf{SK}'_{DL}[l_1] = \big(\{\mathbf{K}'[j,k,l_1]\}_{j,k},\ \{\mathbf{L}'[j,l_1]\}_j,\ \{\mathbf{L}'[j,k,i_1,l_1]\}_{j,k,i_1}, \{\mathbf{L}'_w[j,k,l_1]\}_{j,k}\big),$$

with $j \in \{\ell+2, \ldots, d\}$, $k \in \{1, \ldots, \mu\}$, $i_1 \in \{1, \ldots, \ell+1\}$. Return the delegated private key $\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell,\mathsf{id}_{\ell+1})}[l_1] = (\mathbf{SK}'_D[l_1], \mathbf{SK}'_{DL}[l_1])$.

Return $\{\mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell,\mathsf{id}_{\ell+1})}[l_1]\}_{l_1=1}^n$.

HF.Eval$\big(\mathsf{pms}, \mathsf{mpk}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), X\big)$**:** Given a n-bit input $X = x_1 \ldots x_n \in \{0,1\}^n$, for $i_1 = 1$ to ℓ, parse id_{i_1} as $\mathsf{id}_{i_1}[1] \ldots \mathsf{id}_{i_1}[\mu]$. For $l_1 = 1$ to n, do the following.

1. Compute modified HPE ciphertexts by defining

$$\begin{aligned}\mathbf{C}_{\mathsf{id}}[i_1, l_2, l_1] &= \prod_{i_2=1}^{\mu} \mathbf{C}[i_1, i_2, l_2, l_1]^{\mathsf{id}_{i_1}[i_2]} \\ &= \Big(\mathbf{h}[i_1, 0, l_1]^{\langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1}\rangle \cdot \Delta(l_2,l_1)} \cdot \prod_{i_2=1}^{\mu} \mathbf{h}[i_1, i_2, l_1]^{\mathsf{id}_{i_1}[i_2]}\Big)^{\mathbf{s}[l_2]}\end{aligned}$$

for each $i_1 \in \{1, \ldots, \ell\}$, $l_1, l_2 \in \{1, \ldots, n\}$. The modified ciphertexts are

$$\mathbf{C}_{\mathsf{id}}[l_2, l_1] = \big(\mathbf{J}[l_2], \{\mathbf{C}_{\mathsf{id}}[i_1, l_2, l_1]\}_{i_1=1}^{\ell}\big) \in \mathbb{G}^{\ell+1}. \tag{7}$$

The resulting $\{\mathbf{C}_{id}[l_2, l_1]\}_{l_2,l_1 \in \{1,\ldots,n\}}$ thus form a $n \times n$ matrix of anonymous HIBE ciphertexts for the identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$.

2. Using the vector $X \in \{0,1\}^n$, compute $C_{\mathsf{id},v} = \prod_{l_2=1}^n \mathbf{J}[l_2]^{x_{l_2}} = v^{\langle \mathbf{s}, X\rangle}$, $\mathbf{CT}_{\mathsf{id},w}[l_1] = \prod_{l_2=1}^n \mathbf{C}_w[l_2, l_1]^{x_{l_2}} = \mathbf{w}[l_1]^{\langle s, X\rangle}$ and

$$\begin{aligned}\mathbf{CT}_{\mathsf{id}}[i_1, l_1] &= \prod_{l_2=1}^{n} \mathbf{C}_{\mathsf{id}}[i_1, l_2, l_1]^{x_{l_2}} \\ &= \mathbf{h}[i_1, 0, l_1]^{\mathbf{s}[l_1] \cdot x_{l_1} \cdot \langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1}\rangle} \cdot \Big(\prod_{i_2=1}^{\mu} \mathbf{h}[i_1, i_2, l_1]^{\mathsf{id}_{i_1}[i_2]}\Big)^{\langle \mathbf{s}, X\rangle}\end{aligned} \tag{8}$$

Output

$$C = \big(C_{\mathsf{id},v}, \{\mathbf{CT}_{\mathsf{id},w}[l_1]\}_{l_1=1}^n, \{\mathbf{CT}_{\mathsf{id}}[i_1, l_1]\}_{i_1 \in \{1,\ldots,\ell\}, l_1 \in \{1,\ldots,n\}}\big) \in \mathbb{G}^{n+1+n\times\ell}. \tag{9}$$

HF.Inv$\big(\mathsf{pms}, \mathsf{mpk}, (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathbf{SK}_{(\mathsf{id}_1,\ldots,\mathsf{id}_\ell)}, C\big)$**:** parse the decryption component $\mathbf{SK}_D$ of the private key as a tuple of the form $(\mathbf{D}, \mathbf{D}_w, \mathbf{D}_{\bar{w}}, \{\mathbf{D}_{i_1}\}_{i_1=1}^{\ell})$ and the output C as per (9). Then, for $l_1 = 1$ to n, set $x_{l_1} = 0$ if

$$e(C_{\mathsf{id},v}, \mathbf{D}[l_1]) \cdot e(\mathbf{CT}_{\mathsf{id},w}[l_1], \mathbf{D}_w[l_1])^{-1} \cdot \prod_{i_1=1}^{\ell} e(\mathbf{CT}_{\mathsf{id}}[i_1, l_1], \mathbf{D}_{i_1}[l_1])^{-1} = 1_{\mathbb{G}_T}. \tag{10}$$

Otherwise, set $x_{l_1} = 1$. Eventually, return $X = x_1 \ldots x_n \in \{0,1\}^n$.

From (8), we see that, with overwhelming probability, if there exists $i_1 \in \{1, \ldots, d\}$ such that $\langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1}\rangle \neq 0$, relation (10) is satisfied if and only if $x_{l_1} =$

0. Indeed, in this case, the output (9) is distributed as a vector of n Boneh-Boyen anonymous HIBE ciphertexts. These ciphertexts correspond to the same encryption exponent $\langle \mathbf{s}, X \rangle$ and are generated under n distinct master public keys sharing the same component $v \in \mathbb{G}$.

When the function is prepared for the injective mode, the auxiliary input consists of a vector $\mathbf{y}^{(0)} = [(1, 0, \ldots, 0)| \ldots |(1, 0, \ldots, 0)] \in \mathbb{Z}_p^{d \cdot \mu}$. Since $\mathsf{id}_{i_1}[1] = 1$ for each i_1, this implies injectivity since $\langle \mathbf{y}_{i_1}^{(0)}, \mathsf{id}_{i_1} \rangle \neq 0$ for each $i_1 \in \{1, \ldots, \ell\}$. In the partially lossy mode, a suitable choice of $\mathbf{y}^{(1)}$ ensures that $\langle \mathbf{y}_{i_1}, \mathsf{id}_{i_1} \rangle = 0$ for each $i_1 \in \{1, \ldots, \ell\}$ with non-negligible probability, which leads to high non-injectivity: from (8), we see that (9) only consists of valid HIBE ciphertexts, so that the inversion algorithm always outputs 0^n.

In the full version of the paper [16], we prove that, under the $\mathcal{P}$-BDH$_1$ and DDH$_2$ assumptions, the scheme provides selective security and adaptive security (for a constant number of levels) for appropriate choices of the auxiliary input.

References

1. R. Anderson. Two Remarks on Public Key Cryptology. Invited lecture, *ACM Conference on Computer and Communications Security*, 1997.
2. M. Bellare, A. Boldyreva, A. O'Neill. Deterministic Encryption: Definitional Equivalences and Constructions without Random Oracles. In *Crypto'07*, *LNCS* 4622, 2007.
3. M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, S. Yilek. Hedged Public-Key Encryption: How to Protect against Bad Randomness. In *Asiacrypt'09*, *LNCS* 5912, 2009. Full version available at `http://eprint.iacr.org/2012/220` .
4. M. Bellare, D. Hofheinz, S. Yilek. Possibility and Impossibility Results for Encryption and Commitment Secure under Selective Opening. In *Eurocrypt'09*, *LNCS* 5479, 2009.
5. M. Bellare, E. Kiltz, C. Peikert, B. Waters. Identity-Based (Lossy) Trapdoor Functions and Applications. In *Eurocrypt'12*, *LNCS* 7237, 2012
6. A. Boldyreva, S. Fehr, A. O'Neill. On Notions of Security for Deterministic Encryption, and Efficient Constructions without Random Oracles. In *Crypto'08*, LNCS 5157, 2008.
7. D. Boneh, X. Boyen. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In *Eurocrypt'04*, *LNCS* 3027, 2004.
8. D. Boneh, B. Waters. Conjunctive, Subset, and Range Queries on Encrypted Data. In *4th Theory of Cryptography Conference* (TCC 2007), *LNCS* 4392, 2007.
9. X. Boyen, B. Waters. Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In *Crypto'06*, *LNCS* 4117, 2006.
10. X. Boyen, B. Waters. Shrinking the Keys of Discrete-Log-Type Lossy Trapdoor Functions. In *ACNS'10*, *LNCS* 6123, 2010.
11. R. Canetti, S. Halevi, J. Katz. A Forward-Secure Public-Key Encryption Scheme. In *Eurocrypt'03*, *LNCS* 2656, 2003.
12. R. Canetti, S. Halevi, J. Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In *Eurocrypt'04*, *LNCS* 3027, 2004.
13. D. Cash, D. Hofheinz, E. Kiltz, C. Peikert. Bonsai Trees, or How to Delegate a Lattice Basis. In *Eurocrypt'10*, *LNCS* 6110, 2004.

14. Y. Dodis, L. Reyzin, A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In *Eurocrypt'04*, *LNCS* 3027, 2004.
15. L. Ducas. Anonymity from Asymmetry: New Constructions for Anonymous HIBE. In *CT-RSA'10*, *LNCS* 5985, 2010.
16. A. Escala, J. Herranz, B. Libert, C. Ràfols. Identity-Based Lossy Trapdoor Functions: New Definitions, Hierarchical Extensions, and Implications. Cryptology ePrint Archive: Report 2012/503, 2012.
17. D. Freeman, O. Goldreich, E. Kiltz, A. Rosen, G. Segev. More Constructions of Lossy and Correlation-Secure Trapdoor Functions. In *PKC'10*, *LNCS* 6056, 2010.
18. C. Gentry, A. Silverberg. Hierarchical ID-Based Cryptography. In *Asiacrypt'02*, *LNCS* 2501, 2002.
19. V. Goyal, O. Pandey, A. Sahai, B. Waters. Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. In *ACM CCS'06*, 2006.
20. J. Håstad, R. Impagliazzo, L. Levin, M. Luby. A Pseudorandom Generator from any One-Way Function. *SIAM Journal on Computing*, vol. 28(4), 1999.
21. B. Hemenway, R. Ostrovsky. Lossy Trapdoor Functions from Smooth Homomorphic Hash Proof Systems. In *Electronic Colloquium on Computational Complexity (ECCC)* 16: 127, 2009.
22. D. Hofheinz. All-But-Many Lossy Trapdoor Functions. In *Eurocrypt'12*, *LNCS* 7237, 2012.
23. J. Katz, A. Sahai, B. Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In *Eurocrypt'08*, *LNCS* 4965, 2008.
24. T. Okamoto, K. Takashima. Hierarchical Predicate Encryption for Inner-Products. In *Asiacrypt'09*, *LNCS* 5912, 2009.
25. C. Peikert, B. Waters. Lossy Trapdoor Functions and their Applications. In *STOC'08*, ACM Press, 2008.
26. A. Sahai, B. Waters. Fuzzy Identity-Based Encryption In *Eurocrypt'05*, *LNCS* 3494, 2005.
27. E. Shi, B. Waters. Delegating Capabilities in Predicate Encryption Systems. In *ICALP'08*, *LNCS* 5126, 2008.
28. A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In *Crypto'84*, *LNCS* 196, 1984.
29. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In *Eurocrypt'05*, *LNCS* 3494, 2005.
30. H. Wee. Dual Projective Hashing and its Applications - Lossy Trapdoor Functions and More. In *Eurocrypt'12*, *LNCS* 7237, 2012.
31. X. Xie, R. Xue, R. Zhang. Deterministic Public Key Encryption and Identity-Based Encryption from Lattices in the Auxiliary-Input Setting. In *SCN'12*, *LNCS* 7485, 2012.