All-But-Many Lossy Trapdoor Functions

Dennis Hofheinz

Karlsruhe Institute of Technology, Karlsruhe, Germany

Abstract. We put forward a generalization of lossy trapdoor functions (LTFs). Namely, all-but-many lossy trapdoor functions (ABM-LTFs) are LTFs that are parametrized with tags. Each tag can either be injective or lossy, which leads to an invertible or a lossy function. The interesting property of ABM-LTFs is that it is possible to generate an arbitrary number of lossy tags by means of a special trapdoor, while it is not feasible to produce lossy tags without this trapdoor.
Our definition and construction can be seen as generalizations of all-but-one LTFs (due to Peikert and Waters) and all-but-N LTFs (due to Hemenway et al.). However, to achieve ABM-LTFs (and thus a number of lossy tags which is not bounded by any polynomial), we have to employ some new tricks. Concretely, we give two constructions that use "disguised" variants of the Waters, resp. Boneh-Boyen signature schemes to make the generation of lossy tags hard without trapdoor. In a nutshell, lossy tags simply correspond to valid signatures. At the same time, tags are disguised (i.e., suitably blinded) to keep lossy tags indistinguishable from injective tags.
ABM-LTFs are useful in settings in which there are a polynomial number of adversarial challenges (e.g., challenge ciphertexts). Specifically, building on work by Hemenway et al., we show that ABM-LTFs can be used to achieve selective opening security against chosen-ciphertext attacks. One of our ABM-LTF constructions thus yields the first SO-CCA secure encryption scheme with compact ciphertexts ($\mathbf{O}(1)$ group elements) whose efficiency does not depend on the number of challenges. Our second ABM-LTF construction yields an IND-CCA (and in fact SO-CCA) secure encryption scheme whose security reduction is independent of the number of challenges and decryption queries.

Keywords: lossy trapdoor functions, public-key encryption, selective opening attacks.

1 Introduction

Lossy trapdoor functions. Lossy trapdoor functions (LTFs) have been formalized by Peikert and Waters [30], in particular as a means to construct chosen-ciphertext (CCA) secure public-key encryption (PKE) schemes from lattice assumptions. In a nutshell, LTFs are functions that may be operated with an injective key (in which case a trapdoor allows to efficiently invert the function), or with a lossy key (in which case the function is highly non-injective, i.e., loses information). The key point is that injective and lossy keys are computationally indistinguishable. Hence, in a security proof (say, for a PKE scheme), injective keys can be replaced with lossy keys without an adversary noticing. But once all keys are lossy, a ciphertext does not contain any (significant) information anymore about the encrypted message. There exist quite efficient constructions of LTFs

based on a variety of assumptions (e.g., [30, 7, 10, 20]). Besides, LTFs have found various applications in public-key encryption [22, 7, 6, 5, 23, 19] and beyond [16, 30, 27] (where [16] implicitly uses LTFs to build commitment schemes).

LTFs with tags and all-but-one LTFs. In the context of CCA-secure PKE schemes, it is useful to have LTFs which are parametrized with a tag[1]. In all-but-one LTFs (ABO-LTFs), all tags are injective (i.e., lead to an injective function), except for one single lossy tag. During a proof of CCA security, this lossy tag will correspond to the (single) challenge ciphertext handed to the adversary. All decryption queries an adversary may make then correspond to injective tags, and so can be handled successfully. ABO-LTFs have been defined, constructed, and used as described by Peikert and Waters [30].

Note that ABO-LTFs are not immediately useful in settings in which there is more than one challenge ciphertext. One such setting is the selective opening (SO) security of PKE schemes ([6], see also [11, 18]). Here, an adversary A is presented with a vector of ciphertexts (which correspond to eavesdropped ciphertexts), and gets to choose a subset of these ciphertexts. This subset is then opened for A; intuitively, this corresponds to a number of corruptions performed by A. A's goal then is to find out any nontrivial information about the *unopened* ciphertexts. It is currently not known how to reduce this multi-challenge setting to a single-challenge setting (such as IND-CCA security). In particular, ABO-LTFs are not immediately useful to achieve SO-CCA security. Namely, if we follow the described route to achieve security, we would have to replace all challenge ciphertexts (and only those) with lossy ones. However, an ABO-LTF has only one lossy tag, while there are many challenge ciphertexts.

All-but-N LTFs and their limitations. A natural solution has been given by Hemenway et al. [23], who define and construct all-but-N LTFs (ABN-LTFs). ABN-LTFs have exactly N lossy tags; all other tags are injective. This can be used to equip exactly the challenge ciphertexts with the lossy tags; all other ciphertexts then correspond to injective tags, and can thus be decrypted. Observe that ABN-LTFs encode the set of lossy tags in their key. (That is, a computationally unbounded adversary could always brute-force search which tags lead to a lossy function.) For instance, the construction of [23] embeds a polynomial in the key (hidden in the exponent of group elements) such that lossy tags are precisely the zeros of that polynomial.

Hence, ABN-LTFs have a severe drawback: namely, the space complexity of the keys is at least linear in N. In particular, this affects the SO secure PKE schemes derived in [23]: there is no single scheme that would work in arbitrary protocols (i.e., for arbitrary N). Besides, their schemes quickly become inefficient as N gets larger, since each encryption requires to evaluate a polynomial of degree N in the exponent.

Our contribution: LTFs with many lossy tags. In this work, we define and construct all-but-many LTFs (ABM-LTFs). An ABM-LTF has superpolynomially many lossy tags, which however require a special trapdoor to be found. This is the most crucial difference to ABN-LTFs: with ABN-LTFs, the set of lossy tags is specified initially, at construction time. Our ABM-LTFs have a trapdoor that allows to sample on the fly from a superpolynomially large pool of lossy tags. (Of course, without that trapdoor,

[1] What we call "tag" is usually called "branch." We use "tag" in view of our later construction, in which tags have a specific structure, and cannot be viewed as branches of a (binary) tree.

and even given arbitrarily many lossy tags, another lossy tag is still hard to find.) This in particular allows for ABM-LTF instantiations with compact keys and images whose size is independent of the number of lossy tags.

Our constructions can be viewed as "disguised" variants of the Waters, resp. Boneh-Boyen (BB) signature schemes [33, 8]. Specifically, lossy tags correspond to valid signatures. However, to make lossy and injective tags appear indistinguishable, we have to blind signatures by encrypting them, or by multiplying them with a random subgroup element. We give more details on our constructions below.

A DCR-based construction. Our first construction operates in $\mathrm{Z}_{N^{s+1}}$. (Larger s yield lossier functions. For our applications, $s = 2$ will be sufficient.) A tag consists of two Paillier/Damgård-Jurik encryptions $\mathsf{E}(x) \in \mathrm{Z}_{N^{s+1}}$. At the core of our construction is a variant of Waters signatures over $\mathrm{Z}_{N^{s+1}}$ whose security can be reduced to the problem of computing $\mathsf{E}(ab)$ from $\mathsf{E}(a)$ and $\mathsf{E}(b)$, i.e., of multiplying Paillier/DJ-encrypted messages. This "multiplication problem" may be interesting in its own right. If it is easy, then Paillier/DJ is *fully* homomorphic; if it is infeasible, then we can use it as a "poor man's CDH assumption" in the plaintext domain of Paillier/DJ.

We stress that our construction does not yield a signature scheme; verification of Waters signatures requires a pairing operation, to which we have no equivalent in $\mathrm{Z}_{N^{s+1}}$. However, we will be able to construct a matrix $M \in \mathrm{Z}_{N^{s+1}}^{3\times 3}$ out of a tag, such that the "decrypted matrix" $\widetilde{M} = \mathsf{D}(M) \in \mathrm{Z}_{N^s}^{3\times 3}$ has low rank iff the signature embedded in the tag is valid. Essentially, this observation uses products of plaintexts occurring in the determinant $\det(\widetilde{M})$ to implicitly implement a "pairing over $\mathrm{Z}_{N^{s+1}}$" and verify the signature. Similar techniques to encode arithmetic formulas in the determinant of a matrix have been used, e.g., by [25, 2] in the context of secure computation.

Our function evaluation is now a suitable multiplication of the encrypted matrix M with a plaintext vector $X \in \mathrm{Z}_{N^s}^3$, similar to the one from Peikert and Waters [30]. Concretely, on input X, our function outputs an encryption of the ordinary matrix-vector product $\widetilde{M} \cdot X$. If $\widetilde{M}$ is non-singular, then we can invert this function using the decryption key. If $\widetilde{M}$ has low rank, however, the function becomes lossy. This construction has compact tags and function images; both consist only of a (small) constant number of group elements, and only the public key has $\mathbf{O}(k)$ group elements, where k is the security parameter. Thus, our construction does not scale in the number N of lossy tags.

A pairing-based construction. Our second uses a product group $\mathrm{G}_1 = \langle g_1 \rangle \times \langle h_1 \rangle$ that allows for a pairing. We will implement BB signatures in $\langle h_1 \rangle$, while we blind with elements from $\langle g_1 \rangle$. Consequently, our security proof requires both the Strong Diffie-Hellman assumption (SDH, [8]) in $\langle h_1 \rangle$ and a subgroup indistinguishability assumption.

Tags are essentially matrices $(W_{i,j})_{i,j}$ for $W_{i,j} \in \mathrm{G}_1 = \langle g_1 \rangle \times \langle h_1 \rangle$. Upon evaluation, this matrix is first suitably paired entry-wise to obtain a matrix $(M_{i,j})_{i,j}$ over $\mathrm{G}_T = \langle g_T \rangle \times \langle h_T \rangle$, the pairing's target group. This operation will ensure that (a) $M_{i,j}$ (for $i \neq j$) always lies in $\langle g_T \rangle$, and (b) $M_{i,i}$ lies in $\langle g_T \rangle$ iff the h_1-factor of $W_{i,i}$ constitutes a valid BB signature for the whole tag. With these ideas in mind, we revisit the original matrix-based LTF construction from [30] to obtain a function with trapdoors.

Unfortunately, using the matrix-based construction from [30] results in rather large tags (of size $\mathbf{O}(n^2)$ group elements for a function with domain $\{0,1\}^n$). On the bright side, a number of random self-reducibility properties allow for a security proof whose

reduction quality does *not* degrade with the number N of lossy tags (i.e., challenge ciphertexts) around. Specifically, neither construction nor reduction scale in N.

Applications. Given the work of [23], a straightforward application of our results is the construction of an SO-CCA secure PKE scheme. (However, a slight tweak is required compared to the construction from [23] — see Section 5.3 for details.) Unlike the PKE schemes from [23], both of our ABM-LTFs give an SO-CCA construction that is independent of N, the number of challenge ciphertexts. Moreover, unlike the SO-CCA secure PKE scheme from [19], our DCR-based SO-CCA scheme has compact ciphertexts of $\mathbf{O}(1)$ group elements. Finally, unlike both [23] and [19], our pairing-based scheme has a reduction that does not depend on N and the number of decryption queries (see the full version for details).

As a side effect, our pairing-based scheme can be interpreted as a new kind of CCA secure PKE scheme with a security proof that is tight in the number of challenges and decryption queries. This solves an open problem of Bellare et al. [4], although the scheme should be seen as a (relatively inefficient) proof of concept rather than a practical system. Also, to be fair, we should mention that the SDH assumption we use in our pairing-based ABM-LTF already has a flavor of accommodating multiple challenges: an SDH instance contains polynomially many group elements.

Open problems. An interesting open problem is to find different, and in particular efficient *and* tightly secure ABM-LTFs under reasonable assumptions. This would imply efficient *and* tightly (SO-)CCA-secure encryption schemes. (With our constructions, one basically has to choose between efficiency and a tight reduction.) Also, our pairing-based PKE scheme achieves only indistinguishability-based, but not (in any obvious way) simulation-based SO security [6]. (To achieve simulation-based SO security, a simulator must essentially be able to efficiently explain lossy ciphertexts as encryptions of any given message, see [6, 19].) However, as we demonstrate in case of our DCR-based scheme, in some cases ABM-LTFs can be equipped with an additional "explainability" property that leads to simulation-based SO security (see the full version for details). It would be interesting to find other applications of ABM-LTFs. One reviewer suggested that ABM-LTFs can be used instead of ABO-LTFs in the commitment scheme from Nishimaki et al. [27], with the goal of attaining *reusable* commitments.

Organization. After fixing some notation in Section 2, we proceed to our definition of ABM-LTFs in Section 3. We define and analyze our DCR-based ABM-LTF in Section 4. We then show how ABM-LTFs imply CCA-secure (indistinguishability-based) selective-opening security in Section 5. Due to lack of space, we postpone a detailed description and analysis of our pairing-based ABM-LTF to the full version.

2 Preliminaries

Notation. For $n \in \mathrm{N}$, let $[n] := \{1, \ldots, n\}$. Throughout the paper, $k \in \mathrm{N}$ denotes the security parameter. For a finite set $\mathcal{S}$, we denote by $s \leftarrow \mathcal{S}$ the process of sampling s uniformly from $\mathcal{S}$. For a probabilistic algorithm A, we denote $y \leftarrow A(x; R)$ the process of running A on input x and with randomness R, and assigning y the result. We let $\mathcal{R}_A$ denote the randomness space of A; we require $\mathcal{R}_A$ to be of the

form $\mathcal{R}_A = \{0,1\}^r$. We write $y \leftarrow A(x)$ for $y \leftarrow A(x;R)$ with uniformly chosen $R \in \mathcal{R}_A$, and we write $y_1, \ldots, y_m \leftarrow A(x)$ for $y_1 \leftarrow A(x), \ldots, y_m \leftarrow A(x)$ with fresh randomness in each execution. If A's running time is polynomial in k, then A is called probabilistic polynomial-time (PPT). The statistical distance of two random variables X and Y over some countable domain S is defined as $\mathsf{SD}(X\,;\,Y) := \frac{1}{2}\sum_{s\in S}|\Pr[X=s] - \Pr[Y=s]|$.

Chameleon hashing. A chameleon hash function (CHF, see [26]) is collision-resistant when only the public key of the function is known. However, this collision-resistance can be broken (in a very strong sense) with a suitable trapdoor. We will assume an input domain of $\{0,1\}^*$. We do not lose (much) on generality here, since one can always first apply a collision-resistant hash function on the input to get a fixed-size input.

Definition 1 (Chameleon hash function). *A* chameleon hash function CH *consists of the following PPT algorithms:*

Key generation. $\mathsf{CH.Gen}(1^k)$ *outputs a key* pk_{CH} *along with a trapdoor* td_{CH}.

Evaluation. $\mathsf{CH.Eval}(pk_{\mathsf{CH}}, X; R_{\mathsf{CH}})$ *maps an input* $X \in \{0,1\}^*$ *to an image* Y. *By* R_{CH}*, we denote the randomness used in the process. We require that if* R_{CH} *is uniformly distributed, then so is* Y *(over its respective domain).*

Equivocation. $\mathsf{CH.Equiv}(td_{\mathsf{CH}}, X, R_{\mathsf{CH}}, X')$ *outputs randomness* R'_{CH} *with*

$$\mathsf{CH.Eval}(pk_{\mathsf{CH}}, X; R_{\mathsf{CH}}) = \mathsf{CH.Eval}(pk_{\mathsf{CH}}, X'; R'_{\mathsf{CH}}) \tag{1}$$

for the corresponding key pk_{CH}. *We require that for any* X, X'*, if* R_{CH} *is uniformly distributed, then so is* R'_{CH}.

We require that CH *is* ***collision-resistant*** *in the sense that given* pk_{CH}*, it is infeasible to find* $X, R_{\mathsf{CH}}, X', R'_{\mathsf{CH}}$ *with* $X \neq X'$ *that meet (1). Formally, for every PPT* B*,*

$$\mathsf{Adv}^{\mathsf{cr}}_{\mathsf{CH},B}(k) := \Pr\left[X \neq X' \text{ and (1) holds} \mid (X, R_{\mathsf{CH}}, X', R'_{\mathsf{CH}}) \leftarrow B(1^k, pk_{\mathsf{CH}})\right]$$

is negligible, where $(pk_{\mathsf{CH}}, td_{\mathsf{CH}}) \leftarrow \mathsf{CH.Gen}(1^k)$.

Lossy trapdoor functions. Lossy trapdoor functions (see [30]) are a variant of trapdoor one-way functions. They may be operated in an "injective mode" (which allows to invert the function) and a "lossy mode" in which the function is non-injective. For simplicity, we restrict to an input domain $\{0,1\}^n$ for polynomially bounded $n = n(k) > 0$.

Definition 2 (Lossy trapdoor function). *A* lossy trapdoor function *(LTF)* LTF *with domain* Dom *consists of the following algorithms:*

Key generation. $\mathsf{LTF.IGen}(1^k)$ *yields an evaluation key* ek *and an inversion key* ik.

Evaluation. $\mathsf{LTF.Eval}(ek, X)$ *(with* $X \in \mathsf{Dom}$*) yields an image* Y. *Write* $Y = f_{ek}(X)$.

Inversion. $\mathsf{LTF.Invert}(ik, Y)$ *outputs a preimage* X. *Write* $X = f^{-1}_{ik}(Y)$.

Lossy key generation. $\mathsf{LTF.LGen}(1^k)$ *outputs an evaluation key* ek'.

We require the following:

Correctness. *For all* $(ek, ik) \leftarrow \mathsf{LTF.IGen}(1^k)$, $X \in \mathsf{Dom}$, *it is* $f^{-1}_{ik}(f_{ek}(X)) = X$.

Indistinguishability. *The first output of* $\mathsf{LTF.IGen}(1^k)$ *is indistinguishable from the output of* $\mathsf{LTF.LGen}(1^k)$*, i.e.,*

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},A}(k) := \Pr\left[A(1^k, ek) = 1\right] - \Pr\left[A(1^k, ek') = 1\right]$$

is negligible for all PPT A*, for* $(ek, ik) \leftarrow \mathsf{LTF.IGen}(1^k)$, $ek' \leftarrow \mathsf{LTF.LGen}(1^k)$.

Lossiness. *We say that* LTF *is* ℓ*-lossy if for all possible* $ek' \leftarrow \mathsf{LTF.LGen}(1^k)$*, the image set* $f_{ek'}(\mathsf{Dom})$ *is of size at most* $|\mathsf{Dom}|/2^\ell$*.*

3 Definition of ABM-LTFs

We are now ready to define ABM-LTFs. As already discussed in Section 1, ABM-LTFs generalize ABO-LTFs and ABN-LTFs in the sense that there is a superpolynomially large pool of lossy tags from which we can sample. We require that even given oracle access to such a sampler of lossy tags, it is not feasible to produce a (fresh) non-injective tag. Furthermore, it should be hard to distinguish lossy from injective tags.

Definition 3 (ABM-LTF). *An* all-but-many lossy trapdoor function (ABM-LTF) ABM *with domain* Dom *consists of the following PPT algorithms:*

Key generation. $\mathsf{ABM.Gen}(1^k)$ *yields an evaluation key* ek*, an inversion key* ik*, and a tag key* tk*. The evaluation key* ek *defines a set* $\mathcal{T} = \mathcal{T}_\mathsf{p} \times \{0,1\}^*$ *that contains the disjoint sets of* lossy tags $\mathcal{T}_\mathsf{loss} \subseteq \mathcal{T}$ *and* injective tags $\mathcal{T}_\mathsf{inj} \subseteq \mathcal{T}$*. Tags are of the form* $t = (t_\mathsf{p}, t_\mathsf{a})$*, where* $t_\mathsf{p} \in \mathcal{T}_\mathsf{p}$ *is the* core part *of the tag, and* $t_\mathsf{a} \in \{0,1\}^*$ *is the* auxiliary part *of the tag.*

Evaluation. $\mathsf{ABM.Eval}(ek, t, X)$ *(for* $t \in \mathcal{T}, X \in \mathsf{Dom}$*) produces* $Y =: f_{ek,t}(X)$*.*

Inversion. $\mathsf{ABM.Invert}(ik, t, Y)$ *(with* $t \in \mathcal{T}_\mathsf{inj}$*) outputs a preimage* $X =: f^{-1}_{ik,t}(Y)$*.*

Lossy tag generation. $\mathsf{ABM.LTag}(tk, t_\mathsf{a})$ *takes as input an auxiliary part* $t_\mathsf{a} \in \{0,1\}^*$ *and outputs a core tag* t_p *such that* $t = (t_\mathsf{p}, t_\mathsf{a})$ *is lossy.*

We require the following:

Correctness. *For all possible* $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$*,* $t \in \mathcal{T}_\mathsf{inj}$*, and* $X \in \mathsf{Dom}$*, it is always* $f^{-1}_{ik,t}(f_{ek,t}(X)) = X$*.*

Lossiness. *We say that* ABM *is* ℓ*-lossy if for all possible* $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$*, and all lossy tags* $t \in \mathcal{T}_\mathsf{loss}$*, the image set* $f_{ek,t}(\mathsf{Dom})$ *is of size at most* $|\mathsf{Dom}|/2^\ell$*.*

Indistinguishability. *Even multiple lossy tags are indistinguishable from random tags:*

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM},A}(k) := \Pr\left[A(1^k, ek)^{\mathsf{ABM.LTag}(tk,\cdot)} = 1\right] - \Pr\left[A(1^k, ek)^{\mathcal{O}_{\mathcal{T}_\mathsf{p}}(\cdot)} = 1\right]$$

is negligible for all PPT A*, where* $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$*, and* $\mathcal{O}_\mathcal{T}(\cdot)$ *ignores its input and returns a uniform and independent core tag* $t_\mathsf{p} \leftarrow \mathcal{T}_\mathsf{p}$*.*

Evasiveness. *Non-injective tags are hard to find, even given multiple lossy tags:*

$$\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM},A}(k) := \Pr\left[A(1^k, ek)^{\mathsf{ABM.LTag}(tk,\cdot)} \in \mathcal{T} \setminus \mathcal{T}_\mathsf{inj}\right]$$

is negligible with $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$*, and for any PPT algorithm* A *that never outputs tags obtained through oracle queries (i.e.,* A *never outputs tags* $t = (t_\mathsf{p}, t_\mathsf{a})$*, where* t_p *has been obtained by an oracle query* t_a*).*

On our tagging mechanism. Our tagging mechanism is different from the mechanism from ABO-, resp. ABN-LTFs. In particular, our tag selection involves an auxiliary and a core tag part; lossy tags can be produced for arbitrary auxiliary tags. (Conceptually, this resembles the two-stage tag selection process from Abe et al. [1] in the context of hybrid

encryption.) On the other hand, ABO- and ABN-LTFs simply have fully arbitrary (user-selected) bitstrings as tags.

The reason for our more complicated tagging mechanism is that during a security proof, tags are usually context-dependent and not simply random. For instance, a common trick in the public-key encryption context is the following: upon encryption, choose a one-time signature keypair (v, s), set the tag to the verification key v, and then finally sign the whole ciphertext using the signing key s. This trick has been used numerous times (e.g., [17, 12, 30, 31]) and ensures that a tag cannot be re-used by an adversary in a decryption query. (To re-use that tag, an adversary would essentially have to forge a signature under v.)

However, in our constructions, in particular lossy tags cannot be freely chosen. (This is different from ABO- and ABN-LTFs and stems from the fact that there are superpolynomially many lossy tags.) But as outlined, during a security proof, we would like to embed auxiliary information in a tag, while being able to force the tag to be lossy. We thus divide the tag into an auxiliary part (which can be used to embed, e.g., a verification key for a one-time signature), and a core part (which will be used to enforce lossiness).

4 A DCR-based ABM-LTF

We now construct an ABM-LTF ABMD in rings $\mathrm{Z}_{N^{s+1}}$ for composite N. Domain and codomain of our function will be $\mathrm{Z}_{N^s}^3$ and $(\mathrm{Z}_{N^{s+1}}^*)^3$, respectively. One should have in mind a value of $s \geq 2$ here, since we will prove that ABMD is $((s-1)\log_2(N))$-lossy.

4.1 Setting and assumptions

In the following, let $N = PQ$ for primes P and Q, and fix a positive integer s. Write $\varphi(N) := (P-1)(Q-1)$. We will silently assume that P and Q are chosen from a distribution that depends on the security parameter. Unless indicated otherwise, all computations will take place in $\mathrm{Z}_{N^{s+1}}$, i.e., modulo N^{s+1}. It will be useful to establish the notation $\mathfrak{h} := 1 + N \in \mathrm{Z}_{N^{s+1}}$. We also define algorithms E and D by $\mathsf{E}(x) = r^{N^s}\mathfrak{h}^x$ for $x \in \mathrm{Z}_{N^s}$ and a uniformly and independently chosen $r \in \mathrm{Z}_{N^{s+1}}^*$, and $\mathsf{D}(c) = ((c^{\varphi(N)})^{1/\varphi(N) \bmod N^s} - 1)/N \in \mathrm{Z}_{N^s}$ for $c \in \mathrm{Z}_{N^{s+1}}$. That is, E and D are Paillier/Damgård-Jurik encryption and decryption operations as in [28, 14], so that $\mathsf{D}(r^{N^s}\mathfrak{h}^x) = x$ and $\mathsf{D}(\mathsf{E}(x)) = x$. Moreover, D can be efficiently computed using the factorization of N. We will also apply D to vectors or matrices over $\mathrm{Z}_{N^{s+1}}$, by which we mean component-wise application. We make the following assumptions:

Assumption 1. *The s-Decisional Composite Residuosity (short: s-DCR) assumption holds iff*

$$\mathsf{Adv}_D^{s\text{-}\mathsf{dcr}}(k) := \Pr\left[D(1^k, N, r^{N^s}) = 1\right] - \Pr\left[D(1^k, N, r^{N^s}\mathfrak{h}) = 1\right]$$

is negligible for all PPT D, where $r \leftarrow \mathrm{Z}_{N^{s+1}}^*$ is chosen uniformly.

Assumption 1 is rather common and equivalent to the semantic security of the Paillier [28] and Damgård-Jurik (DJ) [14] encryption schemes. In fact, it turns out that all

s-DCR assumptions are (tightly) equivalent to 1-DCR [14]. Nonetheless, we make s explicit here to allow for a simpler exposition. Also note that Assumption 1 supports a form of random self-reducibility. Namely, given one challenge element $c \in \mathrm{Z}^*_{N^{s+1}}$, it is possible to generate many fresh challenges c_i with the same decryption $\mathsf{D}(c_i) = \mathsf{D}(c)$ by re-randomizing the r^{N^s} part.

Assumption 2. *The No-Multiplication (short: No-Mult) assumption holds iff*

$$\mathsf{Adv}^{\mathsf{mult}}_A(k) := \Pr\left[A(1^k, N, c_1, c_2) = c_* \in \mathrm{Z}^*_{N^2} \textit{ for } \mathsf{D}(c_*) = \mathsf{D}(c_1) \cdot \mathsf{D}(c_2) \bmod N^s\right]$$

is negligible for all PPT A, where $c_1, c_2 \leftarrow \mathrm{Z}^*_{N^2}$ are chosen uniformly.

The No-Mult assumption stipulates that it is infeasible to *multiply* Paillier-encrypted messages. If No-Mult (along with s-DCR and a somewhat annoying technical assumption explained below) hold, then our upcoming construction will be secure. But if the No-Mult problem is easy, then Paillier encryption is fully homomorphic.[2]

The following technical lemma will be useful later on, because it shows how to lift Z_{N^2}-encryptions to $\mathrm{Z}_{N^{s+1}}$-encryptions.

Lemma 1 (Lifting, implicit in [14]). *Let $s \geq 1$ and $\tau : \mathrm{Z}_{N^2} \to \mathrm{Z}_{N^{s+1}}$ be the canonical embedding with $\tau(c \bmod N^2) = c \bmod N^{s+1}$ for $c \in \mathrm{Z}_{N^2}$ interpreted as an integer from $\{0, \ldots, N^2 - 1\}$. Then, for any $c \in \mathrm{Z}^*_{N^2}$, and $X := \mathsf{D}(\tau(c)) \in \mathrm{Z}_{N^s}$ and $x := \mathsf{D}(c) \in \mathrm{Z}_N$, we have $X = x \bmod N$.*

Proof. Consider the canonical homomorphism $\pi : \mathrm{Z}_{N^{s+1}} \to \mathrm{Z}_{N^2}$. Write $\mathrm{Z}^*_{N^{s+1}} = \langle \mathfrak{g}_s \rangle \times \langle \mathfrak{h}_s \rangle$ for some $\mathfrak{g}_s \in \mathrm{Z}^*_{N^{s+1}}$ of order $\varphi(N)$ and $\mathfrak{h}_s := 1 + N \bmod N^{s+1}$. We have $\pi(\langle \mathfrak{g}_s \rangle) = \langle \mathfrak{g}_1 \rangle$ and $\pi(\mathfrak{h}_s^x) = \mathfrak{h}_1^{x \bmod N}$. Since $\pi \circ \hat{\pi} = \mathrm{id}_{\mathrm{Z}_{N^2}}$, this gives $\hat{\pi}(\mathfrak{g}_1^u \mathfrak{h}_1^x) = \mathfrak{g}_s^{u'} \mathfrak{h}_s^{x + x' N}$ for suitable u', x'.

Unfortunately, we need another assumption to exclude certain corner cases:

Assumption 3. *We require that the following function is negligible for all PPT A:*

$$\mathsf{Adv}^{\mathsf{noninv}}_A(k) := \Pr\left[A(1^k, N) = c \in \mathrm{Z}_{N^2} \textit{ such that } 1 < \gcd(\mathsf{D}(c), N) < N\right].$$

Intuitively, Assumption 3 stipulates that it is infeasible to generate Paillier encryptions of "funny messages." Note that actually *knowing* any such message allows to factor N.

4.2 Our construction

Overall idea. The first idea in our construction will be to use the No-Mult assumption as a "poor man's CDH assumption" in order to implement Waters signatures [33] over $\mathrm{Z}_{N^{s+1}}$. Recall that the verification of Waters signatures requires a pairing operation, which corresponds to the multiplication of two Paillier/DJ-encrypted messages

[2] Of course, there is a third, less enjoyable possibility. It is always conceivable that an algorithm breaks No-Mult with low but non-negligible probability. Such an algorithm may not be useful for constructive purposes. Besides, if either s-DCR or the annoying technical assumption do not hold, then our construction may not be secure.

in our setting. We do not have such a multiplication operation available; however, for our purposes, signatures will never actually have to be verified, so this will not pose a problem. We note that the original Waters signatures from [33] are re-randomizable and thus not *strongly* unforgeable. To achieve the evasiveness property from Definition 3, we will thus combine Waters signatures with a chameleon hash function, much like Boneh et al. [9] did to make Waters signatures strongly unforgeable.

Secondly, we will construct 3×3-matrices $M = (M_{i,j})_{i,j}$ over $\mathrm{Z}_{N^{s+1}}$, in which we carefully embed our variant of Waters signatures. Valid signatures will correspond to singular "plaintext matrices" $\widetilde{M} := (\mathsf{D}(M_{i,j}))_{i,j}$; invalid signatures correspond to full-rank matrices $\widetilde{M}$. We will define our ABM-LTF f as a suitable matrix-vector multiplication of M with an input vector $X \in \mathrm{Z}_{N^s}^3$. For a suitable choice of s, the resulting f will be lossy if $\det(\widetilde{M}) = 0$.

Key generation. $\mathsf{ABM.Gen}(1^k)$ first chooses $N = PQ$, and a key pk_{CH} along with trapdoor td_{CH} for a chameleon hash function CH. Finally, $\mathsf{ABM.Gen}$ chooses $a, b \leftarrow \mathrm{Z}_{N^s}$, as well as $k+1$ values $h_i \leftarrow \mathrm{Z}_{N^s}$ for $0 \le i \le k$, and sets

$$A \leftarrow \mathsf{E}(a) \qquad B \leftarrow \mathsf{E}(b) \qquad H_i \leftarrow \mathsf{E}(h_i) \quad (\text{for } 0 \le i \le k)$$
$$ek = (N, A, B, (H_i)_{i=0}^k, pk_{\mathsf{CH}}) \quad ik = (ek, P, Q) \quad tk = (ek, a, b, (h_i)_{i=0}^k, td_{\mathsf{CH}}).$$

Tags. Recall that a tag $t = (t_{\mathsf{p}}, t_{\mathsf{a}})$ consists of a core part t_{p} and an auxiliary part $t_{\mathsf{a}} \in \{0,1\}^*$. Core parts are of the form $t_{\mathsf{p}} = (R, Z, R_{\mathsf{CH}})$ with $R, Z \in \mathrm{Z}_{N^{s+1}}^*$ and randomness R_{CH} for CH. (Thus, random core parts are simply uniform values $R, Z \in \mathrm{Z}_{N^{s+1}}^*$ and uniform CH-randomness.) With t, we associate the chameleon hash value $T := \mathsf{CH.Eval}(pk_{\mathsf{CH}}, (R, Z, t_{\mathsf{a}}))$, and a group hash value $H := H_0 \prod_{i \in T} H_i$, where $i \in T$ means that the i-th bit of T is 1. Let $h := \mathsf{D}(H) = h_0 + \sum_{i \in T} h_i$. Also, we associate with t the matrices

$$M = \begin{pmatrix} Z & A & R \\ B & \mathfrak{h} & 1 \\ H & 1 & \mathfrak{h} \end{pmatrix} \in \mathrm{Z}_{N^{s+1}}^{3\times 3} \qquad \widetilde{M} = \begin{pmatrix} z & a & r \\ b & 1 & 0 \\ h & 0 & 1 \end{pmatrix} \in \mathrm{Z}_{N^s}^{3\times 3}, \tag{2}$$

where $\widetilde{M} = \mathsf{D}(M)$ is the component-wise decryption of M, and $r = \mathsf{D}(R)$ and $z = \mathsf{D}(Z)$. It will be useful to note that $\det(\widetilde{M}) = z - (ab + rh)$. We will call t *lossy* if $\det(\widetilde{M}) = 0$, i.e., if $z = ab + rh$; we say that t is *injective* if $\widetilde{M}$ is invertible.

Lossy tag generation. $\mathsf{ABM.LTag}(tk, t_{\mathsf{a}})$, given $tk = ((N, A, B, (H_i)_i), a, b, (h_i)_{i=0}^k, td_{\mathsf{CH}})$ and an auxiliary tag part $t_{\mathsf{a}} \in \{0,1\}^*$, picks an image T of CH that can later be explained (using td_{CH}) as the image of an arbitrary preimage (R, Z, t_{a}). Let $h := h_0 + \sum_{i \in T} h_i$ and $R \leftarrow \mathsf{E}(r)$ for uniform $r \leftarrow \mathrm{Z}_{N^s}$, and set $Z \leftarrow \mathsf{E}(z)$ for $z = ab + rh$. Finally, let R_{CH} be CH-randomness for which $T = \mathsf{CH.Eval}(pk_{\mathsf{CH}}, (R, Z, t_{\mathsf{a}}))$. Obviously, this yields uniformly distributed lossy tag parts (R, Z, R_{CH}).

Evaluation. $\mathsf{ABM.Eval}(ek, t, X)$, for $ek = (N, A, B, (H_i)_i, pk_{\mathsf{CH}})$, $t = ((R, Z, R_{\mathsf{CH}}), t_{\mathsf{a}})$, and a preimage $X = (X_i)_{i=1}^3 \in \mathrm{Z}_{N^s}^3$, first computes the matrix $M = (M_{i,j})_{i,j}$ as in (2). Then, $\mathsf{ABM.Eval}$ computes and outputs

$$Y := M \circ X := \left(\prod_{j=1}^{3} M_{i,j}^{X_j} \right)_{i=1}^{3}.$$

Note that the decryption $\mathsf{D}(Y)$ is simply the ordinary matrix-vector product $\mathsf{D}(M) \cdot X$.

Inversion and correctness. $\mathsf{ABM.Invert}(ik, t, Y)$, given an inversion key ik, a tag t, and an image $Y = (Y_i)_{i=1}^3$, determines $X = (Y_i)_{i=1}^3$ as follows. First, $\mathsf{ABM.Invert}$ computes the matrices M and $\widetilde{M} = \mathsf{D}(M)$ as in (2), using P, Q. For correctness, we can assume that the tag t is injective, so $\widetilde{M}$ is invertible; let $\widetilde{M}^{-1}$ be its inverse. Since $\mathsf{D}(Y) = \widetilde{M} \cdot X$, $\mathsf{ABM.Invert}$ can retrieve X as $\widetilde{M}^{-1} \cdot \mathsf{D}(Y) = \widetilde{M}^{-1} \cdot \widetilde{M} \cdot X$.

4.3 Security analysis

Theorem 1 (Security of ABMD). *Assume that Assumption 1, Assumption 2, and Assumption 3 hold, that CH is a chameleon hash function, and that $s \geq 2$. Then the algorithms described in Section 4.2 form an ABM-LTF ABMD as per Definition 3.*

We have yet to prove lossiness, indistinguishability, and evasiveness.

Lossiness. Our proof of lossiness loosely follows Peikert and Waters [30]:

Lemma 2 (Lossiness of ABMD). ABMD *is $((s-1)\log_2(N))$-lossy.*

Proof. Assume an evaluation key $ek = (N, A, B, (H_i)_i, pk_{\mathsf{CH}})$, and a lossy tag t, so that the matrix $\widetilde{M}$ from (2) is of rank ≤ 2. Hence, any fixed decrypted image

$$\mathsf{D}(f_{ek,t}(X)) = \mathsf{D}(M \circ X) = \widetilde{M} \cdot X$$

leaves at least one inner product $\langle C, X \rangle \in \mathrm{Z}_{N^s}$ (for $C \in \mathrm{Z}_{N^s}^3$ that only depends on $\widetilde{M}$) completely undetermined. The additional information contained in the encryption randomness of an image $Y = f_{ek,t}(X)$ fixes the components of X and thus $\langle C, X \rangle$ only modulo $\varphi(N) < N$. Thus, for any given image Y, there are at least $\lfloor N^s/\varphi(N) \rfloor \geq N^{s-1}$ possible values for $\langle C, X \rangle$ and thus possible preimages. The claim follows.

Indistinguishability. Observe that lossy tags can be produced without knowledge of the factorization of N. Hence, even while producing lossy tags, we can use the indistinguishability of Paillier/DJ encryptions $\mathsf{E}(x)$. This allows to substitute the encryptions $R = \mathsf{E}(r), Z = \mathsf{E}(z)$ in lossy tags by independently uniform encryptions. This step also makes the CH-randomness independently uniform, and we end up with random tags. We omit the straightforward formal proof and state:

Lemma 3 (Indistinguishability of ABMD). *Given the assumptions from Theorem 1, ABMD is indistinguishable. Concretely, for any PPT adversary A, there exists an s-DCR distinguisher D of roughly the same complexity as A, such that*

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABMD},A}(k) = \mathsf{Adv}^{s\text{-}\mathsf{dcr}}_{D}(k). \tag{3}$$

The tightness of the reduction in (3) stems from the random self-reducibility of s-DCR.

Evasiveness. It remains to prove evasiveness.

Lemma 4 (Evasiveness of ABMD**).** *Given the assumptions from Theorem 1,* ABMD *is evasive. Concretely, for any PPT adversary A that makes at most $Q = Q(k)$ oracle queries, there exist adversaries B, D, and F of roughly the same complexity as A, with*

$$\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABMD},A}(k) \leq \mathbf{O}(kQ(k)) \cdot \mathsf{Adv}^{\mathsf{mult}}_{F}(k) + \mathsf{Adv}^{\mathsf{noninv}}_{E}(k) + \left|\mathsf{Adv}^{s\text{-}\mathsf{dcr}}_{D}(k)\right| + \mathsf{Adv}^{\mathsf{cr}}_{\mathsf{CH},B}(k).$$

At its core, the proof of Lemma 4 adapts the security proof of Waters signatures to $\mathrm{Z}_{N^{s+1}}$. That is, we will create a setup in which we can prepare $Q(k)$ lossy tags (which correspond to valid signatures), and the tag the adversary finally outputs will be interpreted as a forged signature. Crucial to this argument will be a suitable setup of the group hash function $(H_i)_{i=0}^{k}$. Depending on the (group) hash value, we will either be able to create a lossy tag with that hash, or use any lossy tag with that hash to solve a underlying No-Mult challenge. With a suitable setup, we can hope that with probability $\mathbf{O}(1/(kQ(k)))$, $Q(k)$ lossy tags can be created, and the adversary's output can be used to solve an No-Mult challenge. The proof of Lemma 4 is somewhat complicated by the fact that in order to use the collision-resistance of the employed CHF, we have to first work our way towards a setting in which the CHF trapdoor is not used. This leads to a somewhat tedious "deferred analysis" (see [21]) and the s-DCR term in the lemma.

Proof. We turn to the full proof of Lemma 4. Fix an adversary A. We proceed in games. In **Game** 1, $A(ek)$ interacts with an $\mathsf{ABM.LTag}(tk, \cdot)$ oracle that produces core tag parts for lossy tags $t_{\mathsf{p}} = ((R, Z, R_{\mathsf{CH}}), t_{\mathsf{a}})$ that satisfy $z = ab + rh$ for $r = \mathsf{D}(R)$, $z = \mathsf{D}(Z)$, and $h = \mathsf{D}(H)$ with $H = H_0 \prod_{i \in T} H_i$ and $T = \mathsf{CH.Eval}(pk_{\mathsf{CH}}, (R, Z, t_{\mathsf{a}}))$. Without loss of generality, we assume that A makes exactly Q oracle queries, where $Q = Q(k)$ is a suitable polynomial. Let $\mathsf{bad}_{\mathsf{i}}$ denote the event that the output of A in Game i is a lossy tag, i.e., lies in $\mathcal{T}_{\mathsf{loss}}$. By definition,

$$\Pr\left[\mathsf{bad}_1\right] = \Pr\left[A^{\mathsf{ABM.LTag}(tk,\cdot)}(ek) \in \mathcal{T}_{\mathsf{loss}}\right], \tag{4}$$

where the keys ek and tk are generated via $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$.

Getting rid of (chameleon) hash collisions. To describe **Game** 2, let $\mathsf{bad}_{\mathsf{hash}}$ be the event that A finally outputs a tag $t = ((R, Z, R_{\mathsf{CH}}), t_{\mathsf{a}})$ with a CHF hash $T = \mathsf{CH.Eval}((R, Z, t_{\mathsf{a}}); R_{\mathsf{CH}})$ that has already appeared as the CHF hash of an ABM.LTag output (with the corresponding auxiliary tag part input). Now Game 2 is the same as Game 1, except that we abort (and do not raise event bad_2) if $\mathsf{bad}_{\mathsf{hash}}$ occurs. Obviously,

$$\Pr\left[\mathsf{bad}_1\right] - \Pr\left[\mathsf{bad}_2\right] \leq \Pr\left[\mathsf{bad}_{\mathsf{hash}}\right]. \tag{5}$$

It would seem intuitive to try to use CH's collision resistance to bound $\Pr\left[\mathsf{bad}_{\mathsf{hash}}\right]$. Unfortunately, we cannot rely on CH's collision resistance in Game 2 yet, since we use CH's trapdoor in the process of generating lossy tags. So instead, we use a technique called "deferred analysis" [21] to bound $\Pr\left[\mathsf{bad}_{\mathsf{hash}}\right]$. The idea is to forget about the storyline of our evasiveness proof for the moment and develop Game 2 further up to a point at which we can use CH's collision resistance to bound $\Pr\left[\mathsf{bad}_{\mathsf{hash}}\right]$.

This part of the proof largely follows the argument from Lemma 3. Concretely, we can substitute the lossy core tag parts output by ABM.LTag by uniformly random core

tag parts. At this point, CH's trapdoor is no longer required to implement the oracle A interacts with. Hence we can apply CH's collision resistance to bound $\Pr[\mathsf{bad}_{\mathsf{hash}}]$ in this modified game. This also implies a bound on $\Pr[\mathsf{bad}_{\mathsf{hash}}]$ in Game 2: since the occurrence of $\mathsf{bad}_{\mathsf{hash}}$ is obvious from the interaction between A and the experiment, $\Pr[\mathsf{bad}_{\mathsf{hash}}]$ must be preserved across these transformations. We omit the details, and state the result of this deferred analysis:

$$\Pr[\mathsf{bad}_{\mathsf{hash}}] \leq \left|\mathsf{Adv}^{\mathsf{s\text{-}dcr}}_{D}(k)\right| + \mathsf{Adv}^{\mathsf{cr}}_{\mathsf{CH},B}(k) \tag{6}$$

for suitable adversaries D, E, and B. This ends the deferred analysis step, and we are back on track in our evasiveness proof.

Preparing the setup for our reduction. In **Game** 3, we set up the group hash function given by $(H_i)_{i=0}^{k}$ differently. Namely, for $0 \leq i \leq k$, we choose independent $\gamma_i \leftarrow \mathrm{Z}_{N^s}$, and set

$$H_i := A^{\alpha_i}\mathsf{E}(\gamma_i), \qquad \text{so that} \quad h_i := \mathsf{D}(H_i) = \alpha_i a + \gamma_i \bmod N^s \tag{7}$$

for independent $\alpha_i \in \mathrm{Z}$ yet to be determined. Note that this yields an identical distribution of the H_i no matter how concretely we choose the α_i. For convenience, we write $\alpha = \alpha_0 + \sum_{i\in T}\alpha_i$ and $\gamma = \gamma_0 + \sum_{i\in T}\gamma_i$ for a given tag t with associated CH-image T. This in particular implies $h := \mathsf{D}(H) = \alpha a + \gamma$ for the corresponding group hash $H = H_0\prod_{i\in T} H_i$. Our changes in Game 3 are purely conceptual, and so

$$\Pr[\mathsf{bad}_3] = \Pr[\mathsf{bad}_2]\,. \tag{8}$$

To describe **Game** 4, let $t^{(i)}$ denote the i-th lossy core tag part output by $\mathsf{ABM.LTag}$ (including the corresponding auxiliary part $t_{\mathsf{a}}^{(i)}$), and let t^* be the tag finally output by A. Similarly, we denote with $T^{(i)}, \alpha^*$, etc. the intermediate values for the tags output by $\mathsf{ABM.LTag}$ and A. Now let $\mathsf{good}_{\mathsf{setup}}$ be the event that $\gcd(\alpha^{(i)}, N) = 1$ for all i, and that $\alpha^* = 0$. In Game 4, we abort (and do not raise event bad_4) if $\neg\mathsf{good}_{\mathsf{setup}}$ occurs. (In other words, we only continue if each $H^{(i)}$ has an invertible A-component, and if H^* has no A-component.)

Waters [33] implicitly shows that for a suitable distribution of α_i, the probability $\Pr\left[\mathsf{good}_{\mathsf{setup}}\right]$ can be kept reasonably high:

Lemma 5 (Waters [33], Claim 2, adapted to our setting). *In the situation of Game* 4*, there exist efficiently computable distributions α_i, such that for every possible view* view *that A could experience in Game* 4*, we have*

$$\Pr\left[\mathsf{good}_{\mathsf{setup}} \mid \mathsf{view}\right] \geq \mathbf{O}(1/(kQ(k))). \tag{9}$$

This directly implies

$$\Pr[\mathsf{bad}_4] \geq \Pr\left[\mathsf{good}_{\mathsf{setup}}\right] \cdot \Pr[\mathsf{bad}_3]\,. \tag{10}$$

An annoying corner case. Let $\Pr[\mathsf{bad}_{\mathsf{tag}}]$ be the event that A outputs a tag t^* for which $\det(\widetilde{M^*}) = z^* - (ab + r^*h^*)$ is neither invertible nor 0 modulo N. This in

particular means that t^* is neither injective nor lossy. A straightforward reduction to Assumption 3 shows that

$$\Pr[\mathsf{bad}_{\mathsf{tag}}] \leq \mathsf{Adv}_E^{\mathsf{noninv}}(k) \tag{11}$$

for an adversary E that simulates Game 4 and outputs $Z^*/(\mathfrak{h}^{ab} \cdot (R^*)^{h^*}) \bmod N^2$.

The final reduction. We now claim that

$$\Pr[\mathsf{bad}_4] \leq \mathsf{Adv}_F^{\mathsf{mult}}(k) + \Pr[\mathsf{bad}_{\mathsf{tag}}] \tag{12}$$

for the following adversary F on the No-Mult assumption. Our argument follows in the footsteps of the security proof of Waters' signature scheme [33]. Our No-Mult adversary F obtains as input $c_1, c_2 \in \mathrm{Z}_{N^2}$, and is supposed to output $c_* \in \mathrm{Z}_{N^2}$ with $\mathsf{D}(c_1) \cdot \mathsf{D}(c_2) = \mathsf{D}(c_*) \in \mathrm{Z}_N$. In order to do so, F simulates Game 4. F incorporates its own challenge as $A := \tau(c_1)\mathsf{E}(a'N)$ and $B := \tau(c_2)\mathsf{E}(b'N)$ for the embedding τ from Lemma 1 and uniform $a', b' \in \mathrm{Z}_{N^{s-1}}$. This gives uniformly distributed $A, B \in \mathrm{Z}^*_{N^{s+1}}$. Furthermore, by Lemma 1, we have $a = \mathsf{D}(c_1) \bmod N$ and $b = \mathsf{D}(c_2) \bmod N$ for $a := \mathsf{D}(A)$ and $b := \mathsf{D}(B)$. Note that F can still compute all H_i and thus ek efficiently using (7).

We now describe how F constructs lossy tags, as required to implement oracle $\mathcal{T}_{\mathsf{loss}}$ of Game 4. Since the CH trapdoor td_{CH} is under F's control, we can assume a given CH-value T to which we can later map our tag $((R, Z), t_{\mathsf{a}})$. By our changes from Game 4, we can also assume that the corresponding $\alpha = \alpha_0 + \sum_{i \in T} \alpha_i$ is invertible modulo N^s and known. We pick $\delta \leftarrow \mathrm{Z}_{N^s}$, and set

$$R := B^{-1/\alpha \bmod N^s}\mathsf{E}(\delta) \qquad\qquad Z := A^{\alpha\delta}B^{\gamma/\alpha}\mathsf{E}(\gamma\delta).$$

With the corresponding CH-randomness R_{CH}, this yields perfectly distributed lossy tags satisfying

$$\mathsf{D}(A)\cdot\mathsf{D}(B)+\mathsf{D}(R)\cdot\mathsf{D}(H) = ab+(-b/\alpha+\delta)(\alpha a+\gamma) = \alpha\delta a-(\gamma/\alpha)b+\gamma\delta = \mathsf{D}(Z).$$

Note that this generation of lossy tags is *not* possible when $\alpha = 0$.

So far, we have argued that F can simulate Game 4 perfectly for A. It remains to show how F can extract an No-Mult solution out of a tag t^* output by A. Unless $\mathsf{bad}_{\mathsf{tag}}$ occurs or t^* is injective, we have

$$z^* = \mathsf{D}(Z^*) = \mathsf{D}(A) \cdot \mathsf{D}(B) + \mathsf{D}(R^*) \cdot \mathsf{D}(H^*) = ab + r^*h^* \bmod N.$$

Since we abort otherwise, we may assume that $\alpha^* = 0$, so that $z^* = ab + r^*h^* = ab+\gamma^*r^* \bmod N$ for known γ^*. This implies $ab = z^* - \gamma^*r^* \bmod N$, so F can derive and output a Z_{N^2}-encryption of ab as $Z^*/(R^*)^{\gamma^*} \bmod N^2$. This shows (12).

Taking (4)-(12) together shows Lemma 4.

5 Application: selective opening security

5.1 ABM-LTFs with explainable tags

For the application of SOA-CCA security, we need a slight variant of ABM-LTFs. Concretely, we require that values that are revealed during a ciphertext opening can be

explained as uniformly chosen "without ulterior motive," if only their distribution is uniform. (This is called "invertible sampling" by Damgård and Nielsen [15].)

Definition 4 (Efficiently samplable and explainable). *A finite set $\mathcal{S}$ is* efficiently samplable and explainable *if any element of $\mathcal{S}$ can be explained as the result of a uniform sampling. Formally, there are PPT algorithms $\mathsf{Samp}_{\mathcal{S}}$, $\mathsf{Expl}_{\mathcal{S}}$, such that*

1. *$\mathsf{Samp}_{\mathcal{S}}(1^k)$ uniformly samples from $\mathcal{S}$, and*
2. *for any $s \in \mathcal{S}$, $\mathsf{Expl}_{\mathcal{S}}(s)$ outputs random coins for Samp that are uniformly distributed among all random coins R with $\mathsf{Samp}_{\mathcal{S}}(1^k; R) = s$.*

Definition 5 (ABM-LTF with explainable tags). *An ABM-LTF has* explainable tags *if the core part of tags is efficiently samplable and explainable. Formally, if we write $\mathcal{T} = \mathcal{T}_{\mathsf{p}} \times \mathcal{T}_{\mathsf{aux}}$, where $\mathcal{T}_{\mathsf{p}}$ and $\mathcal{T}_{\mathsf{aux}}$ denote the core and auxiliary parts of tags, then $\mathcal{T}_{\mathsf{p}}$ is efficiently samplable and explainable.*

Explainable tags and our ABM-LTFs. Our DCR-based ABM-LTF ABMD has explainable tags, as $\mathrm{Z}^*_{N^{s+1}}$ is efficiently explainable. Concretely, $\mathsf{Samp}_{\mathrm{Z}^*_{N^{s+1}}}$ can choose a uniform $s \leftarrow \mathrm{Z}_{N^{s+1}}$ and test s for invertibility. If s is invertible, we are done; if not, we can factor N and choose a uniform $s' \leftarrow \mathrm{Z}^*_{N^{s+1}}$ directly, using the group order of $\mathrm{Z}^*_{N^{s+1}}$. Similarly, our pairing-based ABM-LTF ABMP has explainable tags as soon as the employed group G_1 is efficiently samplable and explainable. We will also have to explain the CHF randomness R_{CH} in both of our constructions. Fortunately, the CHF randomness of many known constructions [29, 26, 16, 3, 13, 24] consists of uniform values (over an explainable domain), which are efficiently samplable and explainable.

5.2 Selective opening security

PKE schemes. A public-key encryption (PKE) scheme consists of three PPT algorithms $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$. Key generation $\mathsf{PKE.Gen}(1^k)$ outputs a public key pk and a secret key sk. Encryption $\mathsf{PKE.Enc}(pk, msg)$ takes a public key pk and a message msg, and outputs a ciphertext C. Decryption $\mathsf{PKE.Dec}(sk, C)$ takes a secret key sk and a ciphertext C, and outputs a message msg. For correctness, we want $\mathsf{PKE.Dec}(sk, C) = msg$ for all msg, all $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^k)$, and all $C \leftarrow (pk, msg)$. For simplicity, we only consider message spaces $\{0,1\}^k$.

Definition of selective opening security. Following [18, 6, 23], we present a definition for security under selective openings that captures security under adaptive attacks. The definition is indistinguishability-based; it demands that even an adversary that gets to see a vector of ciphertexts cannot distinguish the true contents of the ciphertexts from independently sampled plaintexts.[3] To model adaptive corruptions, our notion also allows the adversary to request "openings" of adaptively selected ciphertexts.

Definition 6 (Efficiently re-samplable). *Let $N = N(k) > 0$, and let dist be a joint distribution over $(\{0,1\}^k)^N$. We say that dist is* efficiently re-samplable *if there is a*

[3] Like previous works, we restrict ourselves to message distributions that allow for an efficient re-sampling. We explain in the full version how to achieve simulation-based selective opening security for *arbitrary* message spaces.

PPT algorithm $\mathsf{ReSamp}_{\mathsf{dist}}$ *such that for any* $\mathcal{I} \subseteq [N]$ *and any partial vector* $\mathbf{msg}'_{\mathcal{I}} := (msg'^{(i)})_{i\in\mathcal{I}} \in (\{0,1\}^k)^{|\mathcal{I}|}$, $\mathsf{ReSamp}_{\mathsf{dist}}(\mathbf{msg}'_{\mathcal{I}})$ *samples from the distribution* dist*, conditioned on* $msg^{(i)} = msg'^{(i)}$ *for all* $i \in \mathcal{I}$.

Definition 7 (IND-SO-CCA security). *A PKE scheme* $\mathsf{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* IND-SO-CCA secure *iff for every polynomially bounded function* $N = N(k) > 0$*, and every stateful PPT adversary* A*, the function*

$$\mathsf{Adv}^{\mathsf{cca\text{-}so}}_{\mathsf{PKE},A}(k) := \Pr\left[\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\mathsf{PKE},A}(k) = 1\right] - \frac{1}{2}$$

is negligible. Here, the experiment $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\mathsf{PKE},A}(k)$ *is defined as follows:*

Experiment $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\mathsf{PKE},A}$
$b \leftarrow \{0,1\}$
$(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^k)$
$(\mathsf{dist}, \mathsf{ReSamp}_{\mathsf{dist}}) \leftarrow A^{\mathsf{PKE.Dec}(sk,\cdot)}(pk)$
$\mathbf{msg}_0 := (msg^{(i)})_{i\in[n]} \leftarrow \mathsf{dist}$
$\mathbf{R} := (R^{(i)})_{i\in[n]} \leftarrow (\mathcal{R}_{\mathsf{PKE.Enc}})^N$
$\mathbf{C} := (C^{(i)})_{i\in[n]} := (\mathsf{PKE.Enc}(pk, msg^{(i)}; R^{(i)}))_{i\in[n]}$
$\mathcal{I} \leftarrow A^{\mathsf{PKE.Dec}(sk,\cdot)}(\mathtt{select}, \mathbf{C})$
$\mathbf{msg}_1 := \mathsf{ReSamp}_{\mathsf{dist}}(\mathbf{msg}_{\mathcal{I}})$
$out_A \leftarrow A^{\mathsf{PKE.Dec}(sk,\cdot)}(\mathtt{output}, (msg^{(i)}, R^{(i)})_{i\in\mathcal{I}}, \mathbf{msg}_b)$
return $(out_A = b)$

We only allow A *that (a) always output efficiently re-samplable distributions* dist *over* $(\{0,1\}^k)^N$ *with corresponding efficient re-sampling algorithms* $\mathsf{ReSamp}_{\mathsf{dist}}$*, (b) never submit a received challenge ciphertext* $C^{(i)}$ *to their decryption oracle* $\mathsf{PKE.Dec}(sk, \cdot)$*, and (c) always produce binary final output* out_A.

This definition can be generalized in many ways, e.g., to more opening phases, or more encryption keys. We focus on the one-phase, one-key case for ease of presentation; our techniques apply equally to a suitably generalized security definitions.

5.3 IND-SO-CCA security from ABM-LTFs

The construction. To construct our IND-SO-CCA secure PKE scheme, we require the following ingredients:

- an LTF $\mathsf{LTF} = (\mathsf{LTF.IGen}, \mathsf{LTF.Eval}, \mathsf{LTF.Invert}, \mathsf{LTF.LGen})$ with domain $\{0,1\}^n$ (as in Definition 2) that is ℓ'-lossy,
- an efficiently explainable ABM-LTF $\mathsf{ABM} = (\mathsf{ABM.Gen}, \mathsf{ABM.Eval}, \mathsf{ABM.Invert}, \mathsf{ABM.LTag})$ with domain[4] $\{0,1\}^n$ and tag set $\mathcal{T} = \mathcal{T}_{\mathsf{p}} \times \mathcal{T}_{\mathsf{aux}}$ (as in Definition 5) that is ℓ-lossy, and

[4] In case of our DCR-based ABM-LTF ABMD, the desired domain $\{0,1\}^n$ must be suitably mapped to ABMD's "native domain" $\mathrm{Z}^3_{N^s}$.

– a family $\mathcal{UH}$ of universal hash functions $h : \{0,1\}^n \to \{0,1\}^k$, so that for any $f : \{0,1\}^n \to \{0,1\}^{\ell'+\ell}$, it is $\mathsf{SD}\left((h, f(X), h(X)) ; (h, f(X), U)\right) = \mathbf{O}(2^{-2k})$, where $h \leftarrow \mathcal{UH}$, $X \leftarrow \{0,1\}^n$, and $U \leftarrow \{0,1\}^k$.

Then, consider the following PKE scheme $\mathsf{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$:

Alg. $\mathsf{PKE.Gen}(1^k)$	**Alg.** $\mathsf{PKE.Enc}(pk, msg)$	**Alg.** $\mathsf{PKE.Dec}(sk, C)$
$(ek', ik') \leftarrow \mathsf{LTF.IGen}(1^k)$	parse $pk =: (ek', ek)$	parse $sk =: (ik', ek)$,
$(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$	$X \leftarrow \{0,1\}^n$	$C =: (\rho, Y', t_\mathsf{p}, Y)$
$pk := (ek', ek)$	$\rho := h(X) \oplus msg$	$X \leftarrow f^{-1}_{ik'}(Y')$
$sk := (ik', ek)$	$Y' := f_{ek'}(X)$	if $Y \neq f_{ek,(t_\mathsf{p},(\rho,Y'))}(X)$
return (pk, sk)	$t_\mathsf{p} := \mathsf{Samp}_{\mathcal{T}_\mathsf{p}}(1^k; R_{t_\mathsf{p}})$	return $\perp$
	$Y := f_{ek,(t_\mathsf{p},(\rho,Y'))}(X)$	$msg := h(X) \oplus \rho$
	$C := (\rho, Y', t_\mathsf{p}, Y)$	return msg
	return C	

The core of this scheme is a (deterministic) double encryption as in [30, 23]. One encryption (namely, Y') is generated using an LTF, and the other (namely, Y) is generated using an ABM-LTF. In the security proof, the LTF will be switched to lossy mode, and the ABM-LTF will be used with lossy tags precisely for the (IND-SO-CCA) challenge ciphertexts. This will guarantee that all challenge ciphertexts will be lossy. At the same time, the evasiveness property of our ABM-LTF will guarantee that no adversary can come up with a decryption query that corresponds to a lossy ABM-LTF tag. As a consequence, we will be able to answer all decryption queries during the security proof.

Relation to the construction of Hemenway et al.. Our construction is almost identical to the one of Hemenway et al. [23], which in turn builds upon the construction of an IND-CCA secure encryption scheme from an all-but-one lossy trapdoor function [30]. However, while we employ ABM-LTFs, [23] employ "all-but-N lossy trapdoor functions" (ABN-LTFs), which are defined similarly to ABM-LTFs, only with the number of lossy tags fixed in advance (to a polynomial value N). Thus, unlike in our schemes, the number of challenge ciphertexts N has to be fixed in advance with [23]. Furthermore, the complexity of the schemes from [23] grows (linearly) in the number N of challenge ciphertexts. On the other hand, ABN-LTFs also allow to explicitly determine all lossy tags in advance, upon key generation. (For instance, all lossy tags can be chosen as suitable signature verification keys or chameleon hash values.) With ABM-LTFs, lossy tags are generated on the fly, through $\mathsf{ABM.LTag}$. This difference is the reason for the auxiliary tag parts in the ABM-LTF definition, cf. Section 3.

Theorem 2. *If* LTF *is an LTF,* ABM *is an efficiently explainable ABM-LTF, and* $\mathcal{UH}$ *is an UHF family as described, then* PKE *is IND-SO-CCA secure. In particular, for every IND-SO-CCA adversary* A *on* PKE *that makes at most* $q = q(k)$ *decryption queries, there exist adversaries* B*,* C*, and* D *of roughly same complexity as* A*, and such that*

$$\left|\mathsf{Adv}^{\mathsf{cca\text{-}so}}_{\mathsf{PKE},A}(k)\right| \leq \left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM},B}(k)\right| + q(k) \cdot \mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM},C}(k) + \left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},D}(k)\right| + \mathbf{O}(2^{-k}).$$

Note that the reduction does not depend on N, the number of challenge ciphertexts. On the other hand, the number of an adversary's decryption queries goes linearly into

the reduction factor. We can get rid of this factor of $q(k)$ in case of our pairing-based ABM-LTF ABMP; see the full version. We also prove Theorem 2 in the full version.

Acknowledgements. The author would like to thank Florian Böhl, Serge Fehr, Eike Kiltz, and Hoeteck Wee for helpful discussions concerning SO-CCA security. The anonymous Crypto 2011 and Eurocrypt 2012 referees, and in particular one Eurocrypt referee have given very useful comments that helped to improve the paper. Jorge Villar pointed me to [32], a result that improves the reduction of our pairing-based ABM-LTF.

References

[1] Masayuki Abe, Rosario Gennaro, and Kaoru Kurosawa. Tag-KEM/DEM: A new framework for hybrid encryption. *Journal of Cryptology*, 21(1):97–130, January 2008.

[2] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In *52nd FOCS*. IEEE Computer Society Press, 2011.

[3] Giuseppe Ateniese and Breno de Medeiros. Identity-based chameleon hash and applications. In Ari Juels, editor, *FC 2004*, volume 3110 of *LNCS*, pages 164–180. Springer, February 2004.

[4] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, May 2000.

[5] Mihir Bellare, Zvika Brakerski, Moni Naor, Thomas Ristenpart, Gil Segev, Hovav Shacham, and Scott Yilek. Hedged public-key encryption: How to protect against bad randomness. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 232–249. Springer, December 2009.

[6] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35. Springer, April 2009.

[7] Alexandra Boldyreva, Serge Fehr, and Adam O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 335–359. Springer, August 2008.

[8] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, April 2008.

[9] Dan Boneh, Emily Shen, and Brent Waters. Strongly unforgeable signatures based on computational Diffie-Hellman. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 229–240. Springer, April 2006.

[10] Xavier Boyen and Brent Waters. Shrinking the keys of discrete-log-type lossy trapdoor functions. In Jianying Zhou and Moti Yung, editors, *ACNS 10*, volume 6123 of *LNCS*, pages 35–52. Springer, June 2010.

[11] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996.

[12] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 207–222. Springer, May 2004.

[13] Benoît Chevallier-Mames and Marc Joye. A practical and tightly secure signature scheme without hash function. In Masayuki Abe, editor, *CT-RSA 2007*, volume 4377 of *LNCS*, pages 339–356. Springer, February 2007.

[14] Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001*, volume 1992 of *LNCS*, pages 119–136. Springer, February 2001.

[15] Ivan Damgård and Jesper Buus Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 432–450. Springer, August 2000.

[16] Ivan Damgård and Jesper Buus Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 581–596. Springer, August 2002.

[17] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *23rd ACM STOC*, pages 542–552. ACM Press, May 1991.

[18] Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. Magic functions. In *40th FOCS*, pages 523–534. IEEE Computer Society Press, October 1999.

[19] Serge Fehr, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Encryption schemes secure against chosen-ciphertext selective opening attacks. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 381–402. Springer, May 2010.

[20] David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 279–295. Springer, May 2010.

[21] Rosario Gennaro and Victor Shoup. A note on an encryption scheme of Kurosawa and Desmedt. Cryptology ePrint Archive, Report 2004/194, 2004. `http://eprint.iacr.org/`.

[22] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[23] Brett Hemenway, Benoit Libert, Rafail Ostrovsky, and Damien Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In *ASIACRYPT*, LNCS. Springer, 2011.

[24] Susan Hohenberger and Brent Waters. Realizing hash-and-sign signatures under standard assumptions. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 333–350. Springer, April 2009.

[25] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *ICALP 2002*, LNCS, pages 244–256. Springer, 2002.

[26] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS 2000*. The Internet Society, February 2000.

[27] Ryo Nishimaki, Eiichiro Fujisaki, and Keisuke Tanaka. Efficient non-interactive universally composable string-commitment schemes. In *ProvSec 2009*, pages 3–18, 2009.

[28] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, May 1999.

[29] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, August 1992.

[30] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 187–196. ACM Press, May 2008.

[31] Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 419–436. Springer, March 2009.

[32] Jorge Villar. An efficient reduction from DDH to the rank problem. 2011.

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