All-But-Many Lossy Trapdoor Functions

Dennis Hofheinz*

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 employ "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.

*Karlsruhe Institute of Technology, Karlsruhe, Germany, Dennis.Hofheinz@kit.edu

1 Introduction

Lossy trapdoor functions. Lossy trapdoor functions (LTFs) have been formalized by Peikert and Waters [32], 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., [32, 7, 10, 20]). Besides, LTFs have found various applications in public-key encryption [22, 7, 5, 4, 23, 18] and beyond [15, 32, 30].

LTFs with tags and all-but-one LTFs. Specifically 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 that an adversary may make then correspond to injective tags, and can thus be handled successfully. ABO-LTFs have been defined, constructed, and used as described by Peikert and Waters [32].

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 ([5], see also [11, 17]). 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. In this setting, ABO-LTFs are not useful to achieve 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, and thus determine the set of lossy tags.) For instance, the construction of [23] embeds a polynomial in the key (hidden in the exponent of group elements) such that the 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, and even given arbitrarily many lossy tags, another lossy tag is

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

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 [37, 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 with a random subgroup element. We give more details on our constructions below.

A DCR-based construction. Our first construction operates in $\mathbb{Z}_{N^{s+1}}$. (Larger s yield lossier functions. For our applications, $s = 2$ will be sufficient.) A tag consists of two Paillier/Damgård-Jurik encryptions $\mathsf{E}(x) \in \mathbb{Z}_{N^{s+1}}$. At the core of our construction is a variant of Waters signatures over $\mathbb{Z}_{N^{s+1}}$ 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 $\mathbb{Z}_{N^{s+1}}$. However, we will be able to construct a matrix $M \in \mathbb{Z}_{N^{s+1}}^{3\times 3}$ out of a tag, such that the "decrypted matrix" $\widetilde{M} = \mathsf{D}(M) \in \mathbb{Z}_{N^s}^{3\times 3}$ 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 $\mathbb{Z}_{N^{s+1}}$" and verify the signature. Similar techniques to encode arithmetic formulas in the determinant of a matrix have been used, e.g., by [24, 2] in the context of secure computation.

Our function evaluation is now a suitable multiplication of the encrypted matrix M with a plaintext vector $X \in \mathbb{Z}_{N^s}^3$, similar to the one from Peikert and Waters [32]. 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 very compact tags and function images; tags and images consist 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. In particular, our construction does not scale in the number N of lossy tags.

A pairing-based construction. Our second construction operates in a product group $\mathbb{G}_1 = \langle g_1 \rangle \times \langle h_1 \rangle$ 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, see [8]) in $\langle h_1 \rangle$ and a suitable subgroup indistinguishability assumption.

Tags are essentially matrices $(W_{i,j})_{i,j}$ for $W_{i,j} \in \mathbb{G}_1 = \langle g_1 \rangle \times \langle h_1 \rangle$. Upon evaluation, this matrix is first suitably paired entry-wise to obtain a matrix $(M_{i,j})_{i,j}$ over $\mathbb{G}_T = \langle g_T \rangle \times \langle h_T \rangle$, 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 [32] to obtain a function with trapdoors.

Unfortunately, using the matrix-based construction from [32] 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, however, a number of random self-reducibility properties allow us to conduct 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 6.3 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 [18], our DCR-based SO-CCA scheme has compact ciphertexts of $\mathbf{O}(1)$ group elements. Finally, unlike both [23] and [18], our pairing-based scheme has a reduction that does not depend on N and the number of decryption

queries (see Section 5.3 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. [3], 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 [5]. (To achieve simulation-based SO security, a simulator must essentially be able to efficiently explain lossy ciphertexts as encryptions of any given message, see [5, 18].) 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 Section 7 for details).

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 present our pairing-based ABM-LTF in Section 5. We then show how ABM-LTFs imply CCA-secure (indistinguishability-based) selective-opening security in Section 6. Finally, we explain how to achieve simulation-based selective-opening security in Section 7.

2 Preliminaries

Notation. For $n \in \mathbb{N}$, let $[n] := \{1, \ldots, n\}$. Throughout the paper, $k \in \mathbb{N}$ 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 [25]) 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 2.1 (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*, the function*

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

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

Lossy trapdoor functions. Lossy trapdoor functions (see [32]) 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 $\{0,1\}^n$, where $n = n(k) > 0$ is a polynomially bounded function.

Definition 2.2 (Lossy trapdoor function)**.** *A* lossy trapdoor function *(LTF)* LTF *with domain* Dom *consists of the following algorithms:*

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

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

Inversion. $\mathsf{LTF.Invert}(ik, Y)$ *outputs a preimage* X*. We 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* $(ek, ik) \leftarrow \mathsf{LTF.IGen}(1^k)$*, and* $X \in \mathsf{Dom}$*, we have* $f^{-1}_{ik}(f_{ek}(X)) = X$*.*

Indistinguishability. *The output of* $\mathsf{LTF.IGen}(1^k)$ *is computationally indistinguishable from the output of* $\mathsf{LTF.LGen}(1^k)$*, i.e., the following function is negligible for all PPT* A*:*

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

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 3.1 (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. $\mathsf{ABM.Eval}(ek, t, X)$ *(with* $t \in \mathcal{T}$ *and* $X \in \mathsf{Dom}$*) outputs an image* $Y =: f_{ek,t}(X)$*.*

Inversion. $\mathsf{ABM.Invert}(ik, t, Y)$ *(with* $t \in \mathcal{T}$*) 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 any PPT algorithm A, where $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$, and $\mathcal{O}_{\mathcal{T}}(\cdot)$ ignores its input and returns a uniformly and independently sampled core tag $t_\mathsf{p} \leftarrow \mathcal{T}_\mathsf{p}$.

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

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

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., [16, 12, 32, 33]) 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. Our solution is to 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 $\mathbb{Z}_{N^{s+1}}$ for composite N. Domain and codomain of our function will be $\mathbb{Z}_{N^s}^3$ and $(\mathbb{Z}_{N^{s+1}}^*)^3$, 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 $\mathbb{Z}_{N^{s+1}}$, i.e., modulo N^{s+1}. It will be useful to establish the notation $\mathfrak{h} := 1 + N \in \mathbb{Z}_{N^{s+1}}$. We also define algorithms E and D by $\mathsf{E}(x) = r^{N^s}\mathfrak{h}^x$ for $x \in \mathbb{Z}_{N^s}$ and a uniformly and independently chosen $r \in \mathbb{Z}_{N^{s+1}}^*$, and $\mathsf{D}(c) = ((c^{\varphi(N)})^{1/\varphi(N) \bmod N^s} - 1)/N \in \mathbb{Z}_{N^s}$ for $c \in \mathbb{Z}_{N^{s+1}}$. That is, E and D are Paillier/Damgård-Jurik encryption and decryption operations as in [31, 13], so that we have $\mathsf{D}(r^{N^s}\mathfrak{h}^x) = x$ and in particular $\mathsf{D}(\mathsf{E}(x)) = x$. Moreover, D can be efficiently computed using the factorization of N. Occasionally, we will apply D to vectors or matrices over $\mathbb{Z}_{N^{s+1}}$, by which we mean component-wise application. We make the following assumptions:

Assumption 4.1. *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 $r \leftarrow \mathbb{Z}^*_{N^{s+1}}$ is chosen uniformly.

Assumption 4.1 is rather common and equivalent to the semantic security of the Paillier [31] and Damgård-Jurik (DJ) [13] encryption schemes. In fact, it turns out that all s-DCR assumptions are (tightly) equivalent to 1-DCR [13]. Nonetheless, we make s explicit here to allow for a simpler exposition. Also note that Assumption 4.1 supports a form of random self-reducibility. Namely, given one challenge element $c \in \mathbb{Z}^*_{N^{s+1}}$, 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 4.2. *We say that the No-Multiplication (short: No-Mult) assumption holds iff*

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

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

The No-Mult assumption stipulates that it is infeasible to *multiply* Paillier-encrypted messages. In a certain sense, this gives a win-win situation. Namely, 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 $\mathbb{Z}_{N^2}$-encryptions to $\mathbb{Z}_{N^{s+1}}$-encryptions.

Lemma 4.3 (Lifting, implicit in [13])**.** *Let $s \geq 1$ and $\tau : \mathbb{Z}_{N^2} \to \mathbb{Z}_{N^{s+1}}$ be the canonical embedding with $\tau(c \bmod N^2) = c \bmod N^{s+1}$ for $c \in \mathbb{Z}_{N^2}$ interpreted as an integer from $\{0, \ldots, N^2-1\}$. Then, for any $c \in \mathbb{Z}^*_{N^2}$, and $X := \mathsf{D}(\tau(c)) \in \mathbb{Z}_{N^s}$ and $x := \mathsf{D}(c) \in \mathbb{Z}_N$, we have $X = x \bmod N$.*

Proof. Consider the canonical homomorphism $\pi : \mathbb{Z}_{N^{s+1}} \to \mathbb{Z}_{N^2}$. Write $\mathbb{Z}^*_{N^{s+1}} = \langle \mathfrak{g}_s \rangle \times \langle \mathfrak{h}_s \rangle$ for some $\mathfrak{g}_s \in \mathbb{Z}^*_{N^{s+1}}$ 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 $\pi \circ \hat{\pi} = \mathrm{id}_{\mathbb{Z}_{N^2}}$, 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 will need another assumption to exclude certain annoying corner cases:

Assumption 4.4. *We require that the following function is negligible for all PPT A:*

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

Intuitively, Assumption 4.4 stipulates that it is infeasible to generate Paillier encryptions of "funny messages." Note that actually *knowing* any such message would allow 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' signature scheme [37] over $\mathbb{Z}_{N^{s+1}}$. Recall that the verification of Waters signatures requires a pairing operation, which corresponds to the multiplication of two Paillier/DJ-encrypted messages 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.

Secondly, we will construct 3×3-matrices $M = (M_{i,j})_{i,j}$ over $\mathbb{Z}_{N^{s+1}}$, in which we carefully embed our variant of Waters signatures. Valid signatures will correspond to singular "plaintext

[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.

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 $X \in \mathbb{Z}_{N^s}^3$. 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 P and Q and sets $N = PQ$. Next, $\mathsf{ABM.Gen}$ chooses a key pk_{CH} along with trapdoor td_{CH} for a chameleon hash function CH. Finally, $\mathsf{ABM.Gen}$ chooses $a, b \leftarrow \mathbb{Z}_{N^s}$, as well as $k+1$ values $h_i \leftarrow \mathbb{Z}_{N^s}$ for $0 \le i \le k$, and sets

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

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 $R, Z \in \mathbb{Z}_{N^{s+1}}^*$ and randomness R_{CH} for CH. (Thus, random core parts are simply uniform values $R, Z \in \mathbb{Z}_{N^{s+1}}^*$ 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

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

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 $r \leftarrow \mathbb{Z}_{N^s}$, 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. $\mathsf{ABM.Eval}(ek, t, X)$, with $ek = (N, A, B, (H_i)_i, pk_{\mathsf{CH}})$, $t = ((R, Z, R_{\mathsf{CH}}), t_{\mathsf{a}})$, and a preimage $X = (X_i)_{i=1}^3 \in \mathbb{Z}_{N^s}^3$, 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 4.5 (Security of ABMD)**.** *Assume that Assumption 4.1, Assumption 4.2, and Assumption 4.4 hold, that* CH *is a chameleon hash function, and that* $s \ge 2$*. Then the algorithms described in Section 4.2 form an ABM-LTF* ABMD *in the sense of Definition 3.1.*

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

Lossiness. Our proof of lossiness loosely follows Peikert and Waters [32]:

Lemma 4.6 (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 $\langle C, X\rangle \in \mathbb{Z}_{N^s}$ (for $C \in \mathbb{Z}_{N^s}^3$ 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 4.7 (Indistinguishability of ABMD)**.** *Given the assumptions from Theorem 4.5,* 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 4.8 (Evasiveness of ABMD)**.** *Given the assumptions from Theorem 4.5,* 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, such that*

$$\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABMD},A}(k) \leq \mathbf{O}(kQ(k)) \cdot \mathsf{Adv}^{\mathsf{mult}}_{F}(k) + \mathsf{Adv}^{\mathsf{noninv}}_{E}(k) + \left|\mathsf{Adv}^{s\text{-}\mathsf{dcr}}_{D}(k)\right| + \mathsf{Adv}^{\mathsf{cr}}_{\mathsf{CH},B}(k). \tag{4}$$

At its core, the proof of Lemma 4.8 adapts the security proof of Waters signatures to $\mathbb{Z}_{N^{s+1}}$. 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 4.8 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 perhaps somewhat surprising s-DCR term in the lemma.

Proof. We turn to the full proof of Lemma 4.8. 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 $\mathsf{bad_i}$ 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,

$$\Pr[\mathsf{bad_1}] = \Pr\left[A^{\mathsf{ABM.LTag}(tk,\cdot)}(ek) \in \mathcal{T}_{\mathsf{loss}}\right], \tag{5}$$

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 some $\mathsf{ABM.LTag}$-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,

$$\Pr[\mathsf{bad}_1] - \Pr[\mathsf{bad}_2] \leq \Pr[\mathsf{bad}_{\mathsf{hash}}]. \tag{6}$$

It would seem intuitive to try to use CH's collision resistance to bound $\Pr[\mathsf{bad}_{\mathsf{hash}}]$. 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 $\Pr[\mathsf{bad}_{\mathsf{hash}}]$. 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 $\Pr[\mathsf{bad}_{\mathsf{hash}}]$.

This part of the proof largely follows the argument from Lemma 5.7. Concretely, we can substitute the lossy core tag parts output by $\mathsf{ABM.LTag}$ 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:

$$\Pr[\mathsf{bad}_{\mathsf{hash}}] \leq \left|\mathsf{Adv}_D^{\text{s-dcr}}(k)\right| + \mathsf{Adv}_{\mathsf{CH},B}^{\mathsf{cr}}(k). \tag{7}$$

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 $\gamma_i \leftarrow \mathbb{Z}_{N^s}$, and set

$$H_i := A^{\alpha_i}\mathsf{E}(\gamma_i), \qquad \text{so that} \quad h_i := \mathsf{D}(H_i) = \alpha_i a + \gamma_i \bmod N^s \tag{8}$$

for independent $\alpha_i \in \mathbb{Z}$ 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

$$\Pr[\mathsf{bad}_3] = \Pr[\mathsf{bad}_2]. \tag{9}$$

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

Lemma 4.9 (Waters [37], 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*

$$\Pr[\mathsf{good}_{\mathsf{setup}} \mid \mathsf{view}] \geq \mathbf{O}(1/(kQ(k))). \tag{10}$$

This directly implies

$$\Pr[\mathsf{bad}_4] \geq \Pr\left[\mathsf{good}_{\mathsf{setup}}\right] \cdot \Pr[\mathsf{bad}_3]. \tag{11}$$

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 4.4 shows that

$$\Pr[\mathsf{bad}_{\mathsf{tag}}] \leq \mathsf{Adv}_E^{\mathsf{noninv}}(k) \tag{12}$$

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

The final reduction. We now claim that

$$\Pr[\mathsf{bad}_4] \leq \mathsf{Adv}_F^{\mathsf{mult}}(k) + \Pr[\mathsf{bad}_{\mathsf{tag}}] \tag{13}$$

for the following adversary F on the No-Mult assumption. Our argument follows in the footsteps of the security proof of Waters' signature scheme [37]. Our No-Mult adversary F obtains as input $c_1, c_2 \in \mathbb{Z}_{N^2}$, and is supposed to output $c_* \in \mathbb{Z}_{N^2}$ with $\mathsf{D}(c_1) \cdot \mathsf{D}(c_2) = \mathsf{D}(c_*) \in \mathbb{Z}_N$. 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 4.3 and uniform $a', b' \in \mathbb{Z}_{N^{s-1}}$. This gives uniformly distributed $A, B \in \mathbb{Z}^*_{N^{s+1}}$. Furthermore, by Lemma 4.3, 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 (8).

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 $\delta \leftarrow \mathbb{Z}_{N^s}$, 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 $\mathbb{Z}_{N^2}$-encryption of ab as $Z^*/(R^*)^{\gamma^*} \bmod N^2$. This shows (13).

Taking (5)-(13) together shows Lemma 4.8. □

5 A pairing-based ABM-LTF

In this section, we give a construction of a pairing-based all-but-many lossy trapdoor function ABMP with domain $\{0,1\}^n$. We will work in a specific group setting, which we describe next.

5.1 Setting and assumptions

Our construction will require groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ of composite order. For the assumptions we will make, it is generally not a problem to publicize this composite order. However, for our subgroup indistinguishability assumption (and only for this assumption), it will be crucial to hide the *factors* of the respective group orders. (In addition, for this assumption, it will be necessary to hide certain subgroup generators.) While the assumption of composite-order groups leads to

comparatively inefficient instantiations, it is not all too uncommon (e.g., [9, 36, 19, 28]). Unique to our setting, however, is the combination of several assumptions of different type over the same platform group(s).

Concretely, we will work in groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ of composite order such that

$$\mathbb{G}_1 = \langle g_1 \rangle \times \langle h_1 \rangle \qquad \mathbb{G}_2 = \langle g_2 \rangle \times \langle h_2 \rangle \qquad \mathbb{G}_T = \langle g_T \rangle \times \langle h_T \rangle \tag{14}$$

for suitable not a priori known $g_1, h_1, g_2, h_2, g_T, h_T$, such that g_1, h_1, g_2, h_2 are of (distinct) unknown prime orders, and so that

$$g_T = \hat{e}(g_1, g_2) \qquad h_T = \hat{e}(h_1, h_2) \qquad \hat{e}(g_1, h_2) = \hat{e}(h_1, g_2) = 1 \tag{15}$$

for a suitable publicly known (asymmetric) pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We furthermore assume that it is possible to uniformly sample from $\mathbb{G}_1$.

Assumption 5.1. *We make the Decisional Diffie-Hellman assumption in $\langle g_1 \rangle$. Namely, we require that for all PPT D, the function*

$$\mathsf{Adv}^{\mathsf{ddh}}_D(k) := \Pr\left[D(1^k, g_1, h_1, g_2, h_2, g_1^a, g_1^b, g_1^{ab}) = 1\right] - \Pr\left[D(1^k, g_1, h_1, g_2, h_2, g_1^a, g_1^b, g_1^r) = 1\right]$$

is negligible, where $a, b, r \leftarrow [\mathrm{ord}(g_1)]$.

We stress that the because the pairing $\hat{e}$ is asymmetric, it does not help in any obvious way in breaking Assumption 5.1.

The random self-reducibility of DDH. A useful property of DDH-like assumptions such as Assumption 5.1 is their random self-reducibility; this has already been noted and used several times (e.g., [35, 29, 34, 3]). These works observe that from one DDH instance (g_1^a, g_1^b, g_1^c) instance (with fixed g_1), we can construct many $(g_1^{b_i}, g_1^{c_i})$ pairs, such that the $g_1^{b_i}$ are independently uniform, and (a) if $c = ab$, then $c_i = ab_i$ for all i, and (b) if c is independently uniform, then so are all c_i. From this, a straightforward hybrid argument shows:

Lemma 5.2. *Let $M = (M_{i,j}) \in \mathbb{G}_1^{n \times n}$ with $M_{i,j} = g_1^{a_i b_j}$ for uniform a_i, b_j, and let $R = (R_{i,j}) \in \mathbb{G}_1^{n \times n}$ with $R_{i,j} = g_1^{r_{i,j}}$ for uniform $r_{i,j}$. Then, for every PPT D', there exists a PPT D with*

$$\Pr\left[D'(1^k, g_1, h_1, g_2, h_2, M)\right] - \Pr\left[D'(1^k, g_1, h_1, g_2, h_2, R)\right] = n \cdot \mathsf{Adv}^{\mathsf{ddh}}_D(k). \tag{16}$$

The factor of n in (16) is rather annoying, and we do not know how to get rid of it.

Assumption 5.3. *We make the following subgroup indistinguishability assumption: we require that for all PPT E, the function*

$$\mathsf{Adv}^{\mathsf{sub}}_E(k) := \Pr\left[E(1^k, g_1, h_1, g_2, g_2^{\widetilde{s}} h_2^{\widehat{s}}, g_1^r) = 1\right] - \Pr\left[E(1^k, g_1, h_1, g_2, g_2^{\widetilde{s}} h_2^{\widehat{s}}, g_1^r \cdot h_1) = 1\right]$$

is negligible, where $r \leftarrow [\mathrm{ord}(g_1)]$, $\widetilde{s} \leftarrow [\mathrm{ord}(g_2)]$, and $\widehat{s} \leftarrow [\mathrm{ord}(h_2)]$.

Note that in this, it is crucial to hide the generator h_2 and the group order of g_1, resp. g_2 and g_T. On the other hand, the element $g_2^{\widetilde{s}} h_2^{\widehat{s}}$ given to E is simply a uniform element from $\mathbb{G}_2$.

Relation to general subgroup decision problems. Subgroup indistinguishability assumptions such as Assumption 5.3 in pairing groups have turned out quite useful in recent years (e.g., [36, 26, 19, 27]). In view of that, Bellare et al. [6] propose *general subgroup decision (GSD) problems* as a generalized way to look at such subgroup indistinguishability assumptions. In a GSD problem for a group with composite order $\prod_{i \in [n]} p_i$, an adversary specifies two subsets $S_0, S_1 \subseteq [n]$, then gets as challenge an element of order $\prod_{i \in S_b} p_i$, and finally must guess b. In the process, the adversary may ask for elements of order p_i, as long as $i \notin (S_0 \setminus S_1) \cup (S_1 \setminus S_0)$. (Note that if $i \in (S_0 \setminus S_1) \cup (S_1 \setminus S_0)$, pairing an element of order p_i with the challenge would trivially solve the GSD problem). While [6] formulate GSD problems for symmetric pairings (so that $\mathbb{G}_1 = \mathbb{G}_2$), Assumption 5.3 can be

cast as a variant of a GSD problem with $n = 2$ in groups with asymmetric pairing. In this case, the adversary would ask for a challenge from $\mathbb{G}_1$ with $S_0 = \{1\}$ and $S_1 = \{1, 2\}$ (corresponding to g_1^r and $g_1^r \cdot h_1$, respectively). The additional elements E gets in Assumption 5.3 do not allow (in an obvious way, using the pairing) to distinguish challenges.

Conversion to prime-order groups. Intuitively, Assumption 5.3 is the only reason why we work in groups of composite order. A natural question is to ask whether we can convert our construction to prime-order groups, which allow for much more efficient instantiations. Freeman [19] gives a generic mechanism for such conversions. However, the generic conversions of [19] require that the pairing is either used in a *projecting* or *canceling* way (but not both). Roughly speaking, *projecting* means that at some point we explicitly use a subgroup order to "exponentiate away" some blinding factor. *Canceling* means that we rely on, e.g., $\hat{e}(g_1, h_2) = 1$. Unfortunately, we use the pairing both in a projecting *and* canceling way, so that we cannot rely on the conversions of [19]. (We note that Meiklejohn et al. [27] present a blind signature scheme with similar properties to demonstrate limitations of Freeman's conversions.) Hence, unfortunately, we do *not* know how to express our construction in a prime-order setting.

Finally, we make an assumption corresponding to the security of the underlying (Boneh-Boyen) signature scheme:

Assumption 5.4. *We make the Strong Diffie-Hellman assumption [8] in $\langle h_1 \rangle$. Namely, we require that for all PPT F and all polynomials $Q = Q(k)$, the function*

$$\mathsf{Adv}_F^{\mathsf{sdh}}(k) := \Pr\left[F(1^k, g_1, h_1, g_2, h_2, \mathrm{ord}(g_1), \mathrm{ord}(h_1), h_2^v, h_1^v, h_1^{(v^2)}, \ldots, h_1^{(v^Q)}) = (e, h_1^{1/(v+e)}) \text{ with } e \in \mathbb{N}\right]$$

is negligible, where $v \leftarrow [\mathrm{ord}(g_1)]$.

5.2 Construction

Let CH be a chameleon hash function. We now describe our ABM-LTF ABMP.

Key generation. $\mathsf{ABM.Gen}(1^k)$ constructs groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ with uniformly distributed generators $g_1, h_1, g_2, h_2, g_T, h_T$ as in (14), along with a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We assume that $\mathsf{ABM.Gen}$ can do so in a way that $\mathrm{ord}(g_T), \mathrm{ord}(h_T)$ are known to $\mathsf{ABM.Gen}$ (but not public, so that it is still realistic to make Assumption 5.3). $\mathsf{ABM.Gen}$ then uniformly chooses exponents $\widehat{v} \leftarrow [\mathrm{ord}(h_T)]$, and $\widetilde{u}, \widetilde{v}, \widetilde{z} \in [\mathrm{ord}(g_T)]$, as well as a chameleon hash function key pk_{CH} along with trapdoor td_{CH}. Then $\mathsf{ABM.Gen}$ sets

$$\begin{aligned} U &= g_2^{\widetilde{u}} h_2 \\ V &= g_2^{\widetilde{v}} h_2^{\widehat{v}} \\ Z &= g_T^{\widetilde{z}} h_T^{-1} \\ ek &= (g_2, U, V, Z, pk_{\mathsf{CH}}) \\ ik &= \mathrm{ord}(g_T) \\ tk &= (g_1, h_1, \mathrm{ord}(g_T), \mathrm{ord}(h_T), \widetilde{u}, \widetilde{v}, \widetilde{z}, \widehat{v}, td_{\mathsf{CH}}). \end{aligned} \tag{17}$$

Tags. Tags are of the form $t = (t_{\mathsf{p}}, t_{\mathsf{a}})$ with an auxiliary tag part $t_{\mathsf{a}} \in \{0,1\}^*$. The core tag part is $t_{\mathsf{p}} = ((W_{i,j})_{i,j \in [n]}, R_{\mathsf{CH}})$ with $W_{i,j} \in \mathbb{G}_1$ and randomness R_{CH} for CH. Random core tag parts are thus uniformly random elements of $\mathbb{G}_1^{n \times n}$ together with uniform randomness for CH. Even though the following decomposition will generally not be known, we write

$$W_{i,j} = g_1^{\widetilde{w_{i,j}}} h_1^{\widehat{w_{i,j}}}. \tag{18}$$

Connected with a tag t as above is the matrix $M = (M_{i,j})_{i,j \in [n]} \in \mathbb{G}_T^{n \times n}$ defined through

$$\begin{aligned} &\forall i, j \in [n], i \neq j: \quad M_{i,j} = \hat{e}(W_{i,j}, g_2) && \left(= g_T^{\widetilde{w_{i,j}}}\right) \\ &\forall i \in [n]: \quad M_{i,i} = \hat{e}(W_{i,i}, U^e V) \cdot Z && \left(= g_T^{\widetilde{w_{i,i}}(\widetilde{v}+e\widetilde{u})+\widetilde{z}} h_T^{\widehat{w_{i,i}}(\widehat{v}+e)-1}\right), \end{aligned} \tag{19}$$

where $e := \mathsf{CH.Eval}(pk_{\mathsf{CH}}, ((W_{i,j})_{i,j}, t_{\mathsf{a}}); R_{\mathsf{CH}})$ is the hash value of the tag.

Lossy and injective tags. A tag t is *injective* iff for all $i \in [n]$, we have $M_{i,i} \in \mathbb{G}_T \setminus \langle g_T \rangle$, i.e., if every $M_{i,i}$ has a nontrivial h_T-factor. Furthermore, t is *lossy* iff there exist $(r_i, s_i)_{i\in[n]}$ such that

$$M_{i,j} = g_T^{r_i s_j}$$

for all $i, j \in [n]$. Using (17,18,19), and assuming $e \neq -\widehat{v}$ for the moment, injectivity is equivalent to $\widehat{w_{i,i}} \neq 1/(\widehat{v}+e) \bmod \mathrm{ord}(h_T)$ for all i. Similarly, lossiness is equivalent to the following three conditions:

1. $\widehat{w_{i,i}} = 1/(\widehat{v}+e) \bmod \mathrm{ord}(h_T)$,
2. $\widetilde{w_{i,j}} = r_i s_j \bmod \mathrm{ord}(g_T)$ for all $i \neq j$, and
3. $\widetilde{w_{i,i}} = (r_i s_i)/(\widetilde{v} + e\widetilde{u}) - \widetilde{z} \bmod \mathrm{ord}(g_T)$ for all $i \in [n]$

for suitable $(r_i, s_i)_{i\in[n]}$. Obviously, the sets of lossy and injective tags are disjoint.

Lossy tag generation. $\mathsf{ABM.LTag}(tk, t_{\mathsf{a}})$, for $tk = (g_1, h_1, \mathrm{ord}(g_T), \mathrm{ord}(h_T), \widetilde{u}, \widetilde{v}, \widetilde{z}, \widehat{v}, td_{\mathsf{CH}})$ and $t_{\mathsf{a}} \in \{0,1\}^*$ uniformly selects exponents $r_i, s_i \leftarrow [\mathrm{ord}(g_T)]$ (for $i \in [n]$). Furthermore, $\mathsf{ABM.LTag}$ selects a random image e of CH so that, using td_{CH}, e can later be explained as an image of an arbitrary preimage. $\mathsf{ABM.LTag}$ then sets

$$\begin{aligned} W_{i,j} &= g_1^{r_i s_j} && (i \neq j) \\ W_{i,i} &= g_1^{(r_i s_i)/(\widetilde{v}+e\widetilde{u}) - \widetilde{z} \bmod \mathrm{ord}(g_T)} h_1^{1/(\widehat{v}+e) \bmod \mathrm{ord}(h_T)} \end{aligned} \tag{20}$$

and generates randomness R_{CH} with $\mathsf{CH.Eval}(pk_{\mathsf{CH}}, ((W_{i,j})_{i,j}, t_{\mathsf{a}}); R_{\mathsf{CH}}) = e$ using td_{CH}. Finally, $\mathsf{ABM.LTag}$ outputs the core tag part $t_{\mathsf{p}} = ((W_{i,j})_{i,j}, R_{\mathsf{CH}})$. By the discussion above, it is clear that the tag $t = (t_{\mathsf{p}}, t_{\mathsf{a}})$ is lossy.

Evaluation. $\mathsf{ABM.Eval}(ek, t, X)$, with $ek = (g_2, U, V, Z, pk_{\mathsf{CH}})$, $t = (((W_{i,j})_{i,j\in[n]}, R_{\mathsf{CH}}), t_{\mathsf{a}})$, and a preimage $X = (X_1, \ldots, X_n) \in \{0,1\}^n$, first computes the matrix M through (19). From there, $\mathsf{ABM.Eval}$ proceeds similarly to the original lossy trapdoor function construction from [32]. Namely, $\mathsf{ABM.Eval}$ computes

$$Y_i = \prod_{j\in[n]} M_{i,j}^{X_j}$$

for $i \in [n]$ and sets $Y := (Y_1, \ldots, Y_n)$. Note that if we write $Y_i = g_T^{\widetilde{y_i}} h_T^{\widehat{y_i}}$ and use the notation from (17,18), we obtain

$$\begin{aligned} \widetilde{y_i} &= \widetilde{w_{i,i}}(\widetilde{v} + e\widetilde{u}) + \widetilde{z} + \sum_{i\neq j} X_j \widetilde{w_{i,j}} \bmod \mathrm{ord}(g_T) \\ \widehat{y_i} &= ((\widehat{v}+e)\widehat{w_{i,i}} - 1)\, X_i \bmod \mathrm{ord}(h_T). \end{aligned} \tag{21}$$

Inversion and correctness. $\mathsf{ABM.Invert}(ik, t, Y)$, given an inversion key $ik = \mathrm{ord}(g_T)$, a tag t, and an image $Y = (Y_1, \ldots, Y_n)$, determines $X = (X_1, \ldots, X_n) \in \{0,1\}^n$ as follows: if $Y_i^{\mathrm{ord}(g_T)} = 1$, then set $X_i = 0$; else set $X_i = 1$. Intuitively, $\mathsf{ABM.Invert}$ thus projects the image elements Y_i onto their h_T-component, and sets $X_i = 1$ iff Y_i has a nontrivial h_T-component.

Given the decomposition (21) of Y_i, we immediately get correctness. Namely, as soon as t is injective, we have $(\widehat{v}+e)\widehat{w_{i,i}} - 1 \neq 0 \bmod \mathrm{ord}(g_T)$ and thus

$$Y_i^{\mathrm{ord}(g_T)} = 1 \iff \widehat{Y_i} = 0 \iff X_i = 0.$$

5.3 Discussion

Before turning to the security analysis of ABMP, we make two comments about ABMP.

Variation with Waters signatures. We could as well have used Waters signatures [37] instead of BB signatures in the construction of ABMP. In that case, there would be two group elements in the tag per element on the diagonal of the matrix W. The derivation of the matrix M would

then proceed as above for elements outside of the diagonal, and would on the diagonal essentially perform the verification of Waters signatures. The details are straightforward, and the most interesting effects would be:
(a) Instead of the SDH assumption, we would use the CDH assumption in $\langle h_1 \rangle$.
(b) Our evasiveness proof would incur a reduction factor of N (the number of issued lossy tags).
(c) The evaluation key would contain a hash function $(H_i)_{i=0}^k$ of linearly many group elements.

Tight (SO-)CCA security proofs with ABMP. As it will turn out, the security reduction of ABMP does not depend on N, the number of used lossy tags. When implementing our SO-CCA secure PKE scheme PKE from Section 6.3 with ABMP, this yields an SO-CCA reduction independent of the number of challenge ciphertexts. However, note that the reduction of Theorem 6.5 still suffers from a factor of q, the number of decryption queries an adversary makes. We will outline how to get rid of this factor using the specific structure of our upcoming evasiveness proof of ABMP.

Namely, the SO-CCA proof incurs a reduction loss of q while ensuring that all decryption queries refer to injective tags. Evasiveness (as in Definition 3.1) only guarantees that an adversary cannot output a *single* non-injective tag. Hence, our SO-CCA proof is forced to a hybrid argument over all q decryption queries during the evasiveness reduction. However, our upcoming evasiveness proof of ABMP can actually verify signatures embedded in tags during the reduction to the SDH assumption. Hence, our proof actually shows – with the same reduction factors – that no adversary can generate any fresh non-injective tags, *even given many tries*. Plugging this proof directly into the SO-CCA proof gives a reduction that is independent of both N and q.

5.4 Security analysis

Theorem 5.5 (Security of ABMP)**.** *Assume that Assumption 5.1, Assumption 5.3, and Assumption 5.4 hold, and that* CH *is a chameleon hash function. Then the algorithms described in Section 5.2 form an ABM-LTF* ABMP *in the sense of Definition 3.1.*

We have already commented on the correctness property; it remains to prove lossiness, indistinguishability, and evasiveness.

Lossiness. Our proof of lossiness proceeds similarly to the one by Peikert and Waters [32]:

Lemma 5.6 (Lossiness of ABMP)**.** ABMP *is* $(n - \log_2(\mathrm{ord}(g_T)))$*-lossy.*

Proof. Assume an evaluation key $ek = (g_2, U, V, Z, pk_{\mathsf{CH}})$ and a tag t. We may assume that t is lossy, so that for the corresponding matrix M given by (19), we have $M_{i,j} = g_T^{r_i s_j}$ for *all* $i, j \in [n]$. Hence, evaluating via $\mathsf{ABM.Eval}(ek, t, X)$ gives $Y = (Y_1, \ldots, Y_n)$ with

$$Y_i = \prod_{j \in [n]} M_{i,j}^{X_j} = g_T^{\sum_{j \in [n]} X_j r_i s_j} = g_T^{r_i \left(\sum_{j \in [n]} X_j s_j \bmod \mathrm{ord}(g_T) \right)}.$$

Thus, Y depends only on $\sum_{j \in [n]} X_j s_j \bmod \mathrm{ord}(g_T)$, and lossiness as claimed follows. □

Indistinguishability. To prove lossy and random tags indistinguishable, we use — not too surprisingly — the DDH and subgroup assumptions. Namely, we first exchange the DDH-related g_1-factors in lossy tags by random factors, and then randomize the now truly blinded h_1-factors. The main challenge in the proof is to prepare essentially the same experiment with different kinds of setup information provided by the respective assumptions.

Lemma 5.7 (Indistinguishability of ABMP)**.** *Given the assumptions from Theorem 5.5,* ABMP *is indistinguishable. Concretely, for any PPT adversary A, there exist adversaries D, and E of roughly the same complexity as A, such that*

$$\left| \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABMP},A}(k) \right| \leq n \cdot \left| \mathsf{Adv}^{\mathsf{ddh}}_{D}(k) \right| + 2 \left| \mathsf{Adv}^{\mathsf{sub}}_{E}(k) \right| + \mathbf{O}(2^{-k}). \tag{22}$$

Proof. Fix a PPT adversary A. We proceed in games. In **Game** 1, $A(ek)$ interacts with an $\mathsf{ABM.LTag}(tk,\cdot)$ oracle that produces lossy core tag parts $t = ((W_{i,j})_{i,j}, R_{\mathsf{CH}})$ as in (20). Without loss of generality, we assume that A makes at most Q oracle queries for a suitable polynomial $Q = Q(k)$. Let out_i denote the output of A in Game i. By definition,

$$\Pr[out_1 = 1] = \Pr\left[A^{\mathsf{ABM.LTag}(tk)}(ek) = 1\right], \tag{23}$$

where the keys ek and tk are generated via $(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$. We will now make modifications to the oracle that A interacts with; we call the refined oracle of Game i $\mathcal{O}_i$.

In **Game** 2, we modify the way the factors $g_1^{(r_i s_i)/(\widetilde{v}+e\widetilde{u})-\widetilde{z} \bmod \mathrm{ord}(g_T)}$ in (20) are computed. Our goal is to avoid an explicit division modulo $\mathrm{ord}(g_T)$. To this end, we set up $r_i = (\widetilde{v} + e\widetilde{u})r'_i$ for uniform r'_i and all $i \in [n]$. This allows to compute

$$g_1^{(r_i s_i)/(\widetilde{v}+e\widetilde{u})-\widetilde{z} \bmod \mathrm{ord}(g_T)} = g_1^{r'_i s_i - \widetilde{z} \bmod \mathrm{ord}(g_T)} = g_1^{r'_i s_i - \widetilde{z}}$$

without $\mathrm{ord}(g_T)$, and does not change the distribution of the evaluation key ek. We get

$$\Pr[out_2 = 1] = \Pr[out_1 = 1]. \tag{24}$$

In **Game** 3, we modify the way the factors $h_1^{1/(\widehat{v}+e) \bmod \mathrm{ord}(h_T)}$ in (20) are computed. Our goal is to avoid an explicit division modulo $\mathrm{ord}(h_T)$. Concretely, we initially choose Q uniform CH-images $e_1, \ldots, e_Q$ and set $E := \prod_{i\in[Q]} \widehat{v} + e_i \in \mathbb{Z}$ for the secret exponent $\widehat{v}$ generated by $\mathsf{ABM.Gen}$. The oracle $\mathcal{O}_3$ will use e_i as the hash value for the ith generated lossy tag. We then set up h_1, h_2, h_T as in Game 2, but with $h_2 := {h'_2}^E$. This can be done, e.g., by first choosing h'_2 as usual, and then setting $h_2 := {h'_2}^E$ and $h_T := \hat{e}(h_1, h_2)$. Note that this does not change the distribution of the evaluation key ek.

Using this setup, oracle $\mathcal{O}_3$ can compute

$$h_1^{1/(\widehat{v}+e_i) \bmod \mathrm{ord}(h_T)} = {h'_2}^{E/(\widehat{v}+e_i) \bmod \mathrm{ord}(h_T)} = {h'_2}^{\prod_{j\neq i}(\widehat{v}+e_j)}$$

and avoid an explicit division modulo $\mathrm{ord}(h_T)$. Our changes in Game 3 are purely conceptual, and hence we get

$$\Pr[out_3 = 1] = \Pr[out_2 = 1]. \tag{25}$$

Note, however, that in Game 3, knowledge of $\mathrm{ord}(g_T)$ and $\mathrm{ord}(h_T)$ is still required in order to choose uniformly random exponents.

In **Game** 4, we finally do without knowledge of $\mathrm{ord}(g_T)$ and $\mathrm{ord}(h_T)$. Namely, we choose all uniform exponents as suitably large integers — e.g., from $[2^{m+2k}]$, where m is an upper bound on the bitlength of $\mathbb{G}_T$-elements. This may result in a (negligibly) skewed distribution of uniform group elements, but allows us to perform Game 4 without knowledge of any group order. We get

$$|\Pr[out_4 = 1] - \Pr[out_3 = 1]| \leq \mathbf{O}(2^{-k}). \tag{26}$$

At this point, we resort to an asymptotic $\mathbf{O}$ notation to avoid the hassle of keeping track of polynomial factors in front of an exponentially small term. However, note that this statistical defect term vanishes exponentially fast as the bitlength of the approximated exponents grows.

In **Game** 5, we modify the way the g_1-factors in (20) are chosen. Now $\mathcal{O}_5$ outputs core tag parts $t_{\mathsf{p}} = ((W_{i,j})_{i,j}, R_{\mathsf{CH}})$ with components $W_{i,j}$ defined according to

$$\begin{aligned} W_{i,j} &= g_1^{\widetilde{w_{i,j}}} && (i \neq j) \\ W_{i,i} &= g_1^{\widetilde{w_{i,i}}} h_2'^{\prod_{j\neq i}(\widehat{v}+e_j)} \end{aligned}$$

for uniformly and independently random exponents $\widetilde{w_{i,j}}$. Hence, the difference to Game 4 is that all g_1-factors of the tag have been randomized, while the h_1-components remain unchanged. A hybrid argument will show

$$|\Pr[out_5 = 1] - \Pr[out_4 = 1]| \leq n \cdot \left|\mathsf{Adv}_D^{\mathsf{ddh}}(k)\right| + \mathbf{O}(2^{-k}) \tag{27}$$

for a suitable DDH adversary D that employs the reduction from Lemma 5.2. (The $\mathbf{O}(2^{-k})$ term in (27) accounts for the statistical skew between a uniform and approximately uniform exponents.)

In **Game** 6, we randomize all h_1-factors in the $W_{i,j}$. Concretely, we let $\mathcal{O}_6$ choose

$$W_{i,j} = g_1^{\widetilde{w_{i,j}}} h_1^{\widehat{w_{i,j}}} \quad (i,j \in [n]),$$

where both the $\widetilde{w_{i,j}}$ and $\widehat{w_{i,j}}$ are uniformly and independently random. Since all h_1-, resp. h_2-factors in Game 5 and Game 6 are blinded by uniform g_1-factors, we obtain

$$|\Pr[out_6 = 1] - \Pr[out_5 = 1]| \leq 2\left|\mathsf{Adv}^{\mathsf{sub}}_E(k)\right| + \mathbf{O}(2^{-k}) \tag{28}$$

for a suitable subgroup distinguisher E. Namely, on input $g_1, h_1, g_2, g_2^{\widetilde{s}} h_2'^{\widehat{s}}, g_1^r h_1^b$ (for $b \in \{0,1\}$), E implicitly sets $h_2 := h_2'^{\widehat{s}}$ and computes the elements U, V, Z of ek by raising $g_2^{\widetilde{s}} h_2'^{\widehat{s}} = g_2^{\widetilde{s}} h_2$ to the appropriate h_2-power and blinding it with a uniform power of g_2. The $W_{i,j}$ are computed from $g_1^r h_1^b$, again by blinding and raising to either the appropriate h_1-power, or a random power. This yields (28). Again, the $\mathbf{O}(2^{-k})$ term in (28) accounts for the statistical skew between uniform and approximately uniform exponents.

In **Game** 7, we let $\mathcal{O}_7$ choose the hash values e not right away, but according to $e := \mathsf{CH.Eval}(pk_{\mathsf{CH}}, ((W_{i,j})_{i,j}, t_{\mathsf{a}}); R_{\mathsf{CH}})$ for uniform and independent randomness R_{CH}. (Note that as of Game 6, e is not used in the construction of the $W_{i,j}$.) By the definition of CH, this change is purely conceptual, so we get

$$\Pr[out_7 = 1] = \Pr[out_6 = 1] \tag{29}$$

Finally, note that $\mathcal{O}_7$ yields uniformly random tags, just like $\mathcal{O}_{\mathcal{T}}$ from the indistinguishability criterion from Definition 3.1. Putting things together, we get

$$\begin{aligned}\left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABMP},A}(k)\right| &= \left|\Pr\left[A^{\mathsf{ABM.LTag}(tk)}(1^k, ek) = 1\right] - \Pr\left[A^{\mathcal{O}_{\mathcal{T}}}(1^k, ek) = 1\right]\right| \\ &= |\Pr[out_1 = 1] - \Pr[out_7 = 1]| \leq n \cdot \left|\mathsf{Adv}^{\mathsf{ddh}}_D(k)\right| + 2\left|\mathsf{Adv}^{\mathsf{sub}}_E(k)\right| + \mathbf{O}(2^{-k}),\end{aligned}$$

which shows (22) as desired. □

Evasiveness. It remains to prove the evasiveness of ABMP.

Lemma 5.8 (Evasiveness of ABMP)**.** *Given the assumptions from Theorem 5.5,* ABMP *is evasive. Concretely, for any PPT adversary A, there exist adversaries B, D, E, and F of roughly the same complexity as A, such that*

$$\left|\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABMP},A}(k)\right| \leq n \cdot \left|\mathsf{Adv}^{\mathsf{ddh}}_D(k)\right| + 2\left|\mathsf{Adv}^{\mathsf{sub}}_E(k)\right| + \mathsf{Adv}^{\mathsf{cr}}_{\mathsf{CH},B}(k) + \mathsf{Adv}^{\mathsf{sdh}}_F(k) + \mathbf{O}(2^{-k}). \tag{30}$$

We first give some intuition before turning to the full proof. First, consider the computation of the matrix $(M_{i,j})_{i,j}$ as in (19). Observe that when only looking at the h_1 factors of the $W_{i,i}$, essentially a verification of n Boneh-Boyen (BB) signatures takes place. In particular, lossy tags correspond to n valid BB signatures. Moreover, in order to generate a non-injective tag, an adversary will have to forge at least one BB signature. Thus, the core of our proof is simply the security proof of the BB signature scheme [8]. However, the proof of Lemma 5.8 is complicated by the fact that in order to use the security 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 perhaps somewhat surprising DDH and subgroup distinguisher terms in the lemma.

Proof. We turn to the full proof of Lemma 5.8. Fix an adversary A. We proceed in games. In **Game** 1, $A(ek)$ interacts with an $\mathsf{ABM.LTag}(tk, \cdot)$ oracle that produces lossy core tag parts $t_{\mathsf{p}} = ((W_{i,j})_{i,j}, R_{\mathsf{CH}})$ as in (20). Without loss of generality, we assume that A makes exactly Q

oracle queries, where $Q = Q(k)$ is a suitable polynomial. Let $\mathsf{bad_i}$ 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,

$$\Pr[\mathsf{bad_1}] = \Pr\left[A^{\mathsf{ABM.LTag}(tk)}(ek) \in \mathcal{T}_{\mathsf{loss}}\right], \tag{31}$$

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

In **Game** 2, we choose all uniform g_1-exponents as suitably large integers (as in the proof of Lemma 5.7). Because this change is merely statistical, we get

$$|\Pr[\mathsf{bad_2}] - \Pr[\mathsf{bad_1}]| \leq \mathbf{O}(2^{-k}). \tag{32}$$

To describe **Game** 3, denote with $\mathsf{bad_{hash}}$ the event that A eventually outputs a tag $t = (((W_{i,j})_{i,j}, R_{\mathsf{CH}}), t_{\mathsf{a}})$ with a hash $e = \mathsf{CH.Eval}(pk_{\mathsf{CH}}, (W_{i,j})_{i,j}; R_{\mathsf{CH}})$ that has already appeared as the hash of some $\mathsf{ABM.LTag}$-output (with attached auxiliary part). Now Game 3 is the same as Game 2, except that we abort (and do not raise event $\mathsf{bad_3}$) if $\mathsf{bad_{hash}}$ occurs. Obviously, we have

$$|\Pr[\mathsf{bad_3}] - \Pr[\mathsf{bad_2}]| \leq \Pr[\mathsf{bad_{hash}}]. \tag{33}$$

It would seem intuitive to try to use CH's collision resistance to bound $\Pr[\mathsf{bad_{hash}}]$. Unfortunately, we cannot rely on CH's collision resistance in Game 3 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 $\Pr[\mathsf{bad_{hash}}]$. The idea is to forget about the storyline of our evasiveness proof for the moment and develop Game 3 further up to a point at which we can use CH's collision resistance to bound $\Pr[\mathsf{bad_{hash}}]$.

This part of the proof largely follows the proof of Lemma 5.7. Concretely, we can substitute the lossy tags output by $\mathsf{ABM.LTag}$ step by step by (approximately) uniformly random tags. 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 $\Pr[\mathsf{bad_{hash}}]$ in this modified game. This also implies a bound on $\Pr[\mathsf{bad_{hash}}]$ in Game 3: since the occurrence of $\mathsf{bad_{hash}}$ is obvious from the interaction between A and the experiment, $\Pr[\mathsf{bad_{hash}}]$ must be preserved across these transformations. We omit the details, and state the result of this deferred analysis:

$$\Pr[\mathsf{bad_{hash}}] \leq n \cdot \left|\mathsf{Adv}^{\mathsf{ddh}}_D(k)\right| + 2\left|\mathsf{Adv}^{\mathsf{sub}}_E(k)\right| + \mathsf{Adv}^{\mathsf{cr}}_{\mathsf{CH},B}(k) + \mathbf{O}(2^{-k}) \tag{34}$$

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

We now claim that

$$\Pr[\mathsf{bad_3}] = \mathsf{Adv}^{\mathsf{sdh}}_F(k) \tag{35}$$

for the following adversary F on the Strong Diffie-Hellman assumption (i.e., Assumption 5.4). Our reduction follows in the footsteps of the security proof of the Boneh-Boyen signature scheme from [8]. Recall that F obtains as input elements $g_1, h'_1, g_2, h_2, h_2^{\widehat{v}}, {h'_1}^{\widehat{v}}, {h'_1}^{(\widehat{v}^2)} \ldots, {h'_1}^{\widehat{v}^Q}$ for an unknown uniform exponent $\widehat{v}$, as well as $\mathrm{ord}(g_1)$ and $\mathrm{ord}(h'_1)$, and is supposed to output a tuple $(e, {h'_1}^{1/(\widehat{v}+e)})$. First, F chooses Q uniform CH-images $e_1, \ldots, e_Q$ and sets up the degree-Q polynomial

$$f(X) := \prod_{i \in [Q]} X + e_i \quad \in \quad \mathbb{Z}_{\mathrm{ord}(h'_1)}[X]$$

with zeros $-e_i$. Then F constructs $h_1 := {h'_1}^{f(\widehat{v})}$ using the coefficients of $f(X)$ and the elements ${h'_1}^{\widehat{v}^j}$ from its input. Note that now, F is able to efficiently compute roots

$$h_1^{1/(\widehat{v}+e_i)} = {h'_1}^{f(\widehat{v})/(\widehat{v}+e_i)} = {h'_1}^{\prod_{j \neq i} \widehat{v}+e_j}$$

by computing the coefficients of the polynomial $f_i(X) = \prod_{j \neq i} X + e_i \in \mathbb{Z}_{\mathrm{ord}(h'_1)}[X]$ and then constructing ${h'_1}^{f_i(\widehat{v})}$ from the ${h'_1}^{\widehat{v}^i}$.

This allows F to simulate Game 3, using an evaluation key $ek = (g_2, U, V = g_2^{\widetilde{v}} h_2^{\widehat{v}}, Z, pk_{\mathsf{CH}})$ (for uniform $\widetilde{v}$) and tag hash values e_i for the answers to F's oracle queries. Say that F generates a tag $t = ((W_{i,j})_{i,j}, R_{\mathsf{CH}})$ as output. By our change from Game 3, we can assume that the hash $e := \mathsf{CH.Eval}(pk_{\mathsf{CH}}, ((W_{i,j})_{i,j}, t_{\mathsf{a}}); R_{\mathsf{CH}})$ is different from all e_i. In particular, the polynomials $f(X)$ and $X + e$ are coprime, so that we can write $1 = \phi \cdot f(X) + \rho(X) \cdot (X + e)$ for suitable $\phi \in \mathbb{Z}_{\mathrm{ord}(h_1')}$ and $\rho(X) \in \mathbb{Z}_{\mathrm{ord}(h_1')}[X]$ with $\deg(\rho(X)) = Q - 1$. Thus, a nontrivial root of h_1' can be computed as

$$h_1'^{\frac{1}{\widehat{v}+e}} = h_1'^{\frac{\phi \cdot f(\widehat{v}) + \rho(\widehat{v}) \cdot (\widehat{v}+e)}{\widehat{v}+e}} = \left(h_1^{\frac{1}{\widehat{v}+e}}\right)^{\phi} \cdot h_1'^{\rho(\widehat{v})},$$

where $h_1'^{\rho(\widehat{v})}$ can be computed from the $h_1'^{(\widehat{v}^i)}$, and $h_1^{\frac{1}{\widehat{v}+e}}$ is extracted from F's output as follows. Namely, F checks for all indices $i \in [n]$ whether $\hat{e}(W_{i,i}, h_2^{\widehat{v}+e}) \stackrel{?}{=} g_T$. If this is the case for, say, i, then

$$\left(W_{i,i}^{\mathrm{ord}(g_1)}\right)^{\mathrm{ord}(g_1)^{-1} \bmod \mathrm{ord}(h_1')} = h_1^{\frac{1}{\widehat{v}+e}}.$$

If $\hat{e}(W_{i,i}, h_2^{\widehat{v}+e}) \neq g_T$ for all i, then the tag is injective, and F fails.

By this setup, F succeeds to compute $h_1'^{1/(\widehat{v}+e)}$ whenever $\mathsf{bad_3}$ occurs, so we obtain (35). Summing up, we get (30) as desired. □

6 Application: selective opening security

6.1 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 [14].)

Definition 6.1 (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 6.2 (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, since $\mathbb{Z}^*_{N^{s+1}}$ is efficiently samplable and explainable. Concretely, $\mathsf{Samp}_{\mathbb{Z}^*_{N^{s+1}}}$ would choose a uniform $s \leftarrow \mathbb{Z}_{N^{s+1}}$ and test s for invertibility. If s is invertible, we are done; if not, we can factor N and choose a uniform $s' \leftarrow \mathbb{Z}^*_{N^{s+1}}$ directly, using the group order of $\mathbb{Z}^*_{N^{s+1}}$. Similarly, our pairing-based ABM-LTF ABMP has explainable tags as soon as the employed group $\mathbb{G}_1$ is efficiently samplable and explainable.[3]

6.2 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

[3]Technically, we will also have to explain the CHF randomness R_{CH} in both of our constructions. However, we can assume without loss of generality that the CHF randomness is simply a uniformly distributed and suitably long bitstring, which makes it efficiently samplable and explainable.

key sk. Encryption $\mathsf{PKE.Enc}(pk, M)$ takes a public key pk and a message M, and outputs a ciphertext C. Decryption $\mathsf{PKE.Dec}(sk, C)$ takes a secret key sk and a ciphertext C, and outputs a message M. For correctness, we want $\mathsf{PKE.Dec}(sk, C) = M$ for all M, all $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^k)$, and all $C \leftarrow (pk, M)$. For simplicity, we only consider PKE schemes that allow messages $M \in \{0,1\}^k$.

Definition of selective opening security. Following [17, 5, 23], we present a definition for security under selective openings that captures security of an encryption scheme 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.[4] To model adaptive corruptions, our notion also allows the adversary to request "openings" of adaptively selected ciphertexts.

Definition 6.3 (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 PPT algorithm* $\mathsf{ReSamp}_{\mathsf{dist}}$ *such that for any $\mathcal{I} \subseteq [N]$ and any partial vector $\mathbf{M}'_{\mathcal{I}} := (M'^{(i)})_{i \in \mathcal{I}} \in (\{0,1\}^k)^{|\mathcal{I}|}$, $\mathsf{ReSamp}_{\mathsf{dist}}(\mathbf{M}'_{\mathcal{I}})$ samples from the distribution* dist*, conditioned on $M^{(i)} = M'^{(i)}$ for all $i \in \mathcal{I}$.*

Definition 6.4 (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 $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\mathsf{PKE},A}$
- $b \leftarrow \{0,1\}$
- $(pk, sk) \leftarrow \mathsf{PKE.Gen}(1^k)$
- $(\mathsf{dist}, \mathsf{ReSamp}_{\mathsf{dist}}) \leftarrow A^{\mathsf{PKE.Dec}(sk,\cdot)}(pk)$
- $\mathbf{M}_0 := (M^{(i)})_{i \in [n]} \leftarrow \mathsf{dist}$
- $\mathbf{R} := (R^{(i)})_{i \in [n]} \leftarrow (\mathcal{R}_{\mathsf{PKE.Enc}})^N$
- $\mathbf{C} := (C^{(i)})_{i \in [n]} := (\mathsf{PKE.Enc}(pk, M^{(i)}; R^{(i)}))_{i \in [n]}$
- $\mathcal{I} \leftarrow A^{\mathsf{PKE.Dec}(sk,\cdot)}(\mathtt{select}, \mathbf{C})$
- $\mathbf{M}_1 := \mathsf{ReSamp}_{\mathsf{dist}}(\mathbf{M}_{\mathcal{I}})$
- $out_A \leftarrow A^{\mathsf{PKE.Dec}(sk,\cdot)}(\mathtt{output}, (M^{(i)}, R^{(i)})_{i \in \mathcal{I}}, \mathbf{M}_b)$
- return $out_A \oplus b$

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 any 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. Most prominently, one could ask for more opening phases, or more encryption keys. We have focused on the one-phase, one-key case for ease of presentation; however, our techniques apply equally to a suitably generalized security definitions.

6.3 IND-SO-CCA security from ABM-LTFs

Relation to the construction of Hemenway et al.. We are now ready to construct an IND-SO-CCA secure PKE scheme. Our construction will be almost identical to the one of Hemenway et al. [23], which in turn builds upon the construction of an IND-CCA secure encryption scheme

[4] Like previous works that prove *indistinguishability-based* selective opening security, we will restrict ourselves to message distributions that allow for an efficient such a re-sampling. However, we comment in Section 7 on how to achieve simulation-based selective opening security for *arbitrary* message spaces.

from an all-but-one lossy trapdoor function [32]. 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 $\mathsf{ABM.LTag}$. Because of this difference, the introduction of auxiliary tags in the definition of ABM-LTFs was necessary, as explained in Section 3.

The construction. 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 2.2) 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[5] $\{0,1\}^n$ and tag set $\mathcal{T} = \mathcal{T}_{\mathsf{p}} \times \mathcal{T}_{\mathsf{aux}}$ (as in Definition 6.2) that is ℓ-lossy, and
- a family $\mathcal{UH}$ of universal hash functions $h : \{0,1\}^n \to \{0,1\}^k$, such that for any $f : \{0,1\}^n \to \{0,1\}^{\ell'+\ell}$,

$$\mathsf{SD}\left((h, f(X), h(X)) \,;\, (h, f(X), U)\right) = \mathbf{O}(2^{-2k}),$$

  where $h \leftarrow \mathcal{UH}$, $X \leftarrow \{0,1\}^n$, and $U \leftarrow \{0,1\}^k$.

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

Algorithm $\mathsf{PKE.Gen}(1^k)$	**Algorithm** $\mathsf{PKE.Enc}(pk, M)$	**Algorithm** $\mathsf{PKE.Dec}(sk, C)$
$(ek', ik') \leftarrow \mathsf{LTF.IGen}(1^k)$	parse $pk =: (ek', ek)$	parse $sk =: (ik', ek)$,
$(ek, ik, tk) \leftarrow \mathsf{ABM.Gen}(1^k)$	$X \leftarrow \{0,1\}^n$	$C =: (\rho, Y', t_{\mathsf{p}}, Y)$
$pk := (ek', ek)$	$\rho := h(X) \oplus M$	$X \leftarrow f^{-1}_{ik'}(Y')$
$sk := (ik', ek)$	$Y' := f_{ek'}(X)$	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 $\perp$
	$Y := f_{ek,(t_{\mathsf{p}},(\rho,Y'))}(X)$	$M := h(X) \oplus \rho$
	$C := (\rho, Y', t_{\mathsf{p}}, Y)$	return M
	return C	

The core of this scheme is a (deterministic) double encryption as in [32, 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 construction of our tags for the 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.

Theorem 6.5. *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*

$$\left|\mathsf{Adv}^{\mathsf{cca\text{-}so}}_{\mathsf{PKE},A}(k)\right| \leq \left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM},B}(k)\right| + q(k) \cdot \mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM},C}(k) + \left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},D}(k)\right| + \mathbf{O}(2^{-k}). \tag{36}$$

Note that the reduction factors do not depend on N, i.e., the number of challenge ciphertexts. On the other hand, the number of an adversary's decryption queries goes linearly into the reduction factor. (However, we can get rid of this factor of $q(k)$ in case of our pairing-based ABM-LTF ABMP; see Section 5.3 for details.)

We now turn to the proof of Theorem 6.5.

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

Proof. The proof largely follows [32, 23]. Assume $n = n(k) > 0$ and an IND-SO-CCA adversary A that makes exactly q decryption queries, where $q = q(k)$ is a suitable polynomial. We proceed in games, and start with the real IND-SO-CCA experiment $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\mathsf{PKE},A}$ as **Game** 1. In this game, A receives a ciphertext vector $\mathbf{C} := (C^{(i)})_{i\in[n]} := (\mathsf{PKE.Enc}(pk, M^{(i)}; R^{(i)}))_{i\in[n]}$ for a message vector $\mathbf{M} := (M^{(i)})_{i\in[n]}$ sampled from an adversarially chosen message distribution dist. A then chooses a subset $\mathcal{I} \subseteq [n]$ and gets openings $(M^{(i)}, R^{(i)})_{i\in\mathcal{I}}$ of all ciphertexts in $\mathcal{I}$, along with the whole message vector $\mathbf{M}$. In this, each ciphertext $C^{(i)}$ is of the form $(\rho, Y', t_\mathsf{p}, Y)$, and generated according to $\mathsf{PKE.Enc}$. This means that t_p is generated as a random tag part, using $\mathsf{Samp}_{\mathcal{T}_\mathsf{p}}(1^k; R_{t_\mathsf{p}})$. If we denote with out_i the output of Game i, we get

$$\Pr[out_1 = 1] - \frac{1}{2} = \mathsf{Adv}^{\mathsf{cca\text{-}so}}_{\mathsf{PKE},A}(k). \tag{37}$$

Say that a tag $(t_\mathsf{p}, (\rho, Y'))$ in one of A's decryption queries $(\rho, Y', t_\mathsf{p}, Y)$ is *copied* if it has occurred already in one of the challenge ciphertexts $C^{(i)}$. In **Game** 2, we reject (with $\perp$) all copied tags. Note that since A is not allowed to ask for the decryption of challenge ciphertexts, we may assume that $C^{(i)} = (\rho, Y', t_\mathsf{p}, \hat{Y})$ for $\hat{Y} \neq Y$ in such a case. By construction of the challenge ciphertexts, however, we have

$$Y \neq \hat{Y} = f_{ek,(t_\mathsf{p},(\rho,Y'))}(X)$$

for $X := f^{-1}_{ik'}(Y')$. Hence, decryption queries with copied tags would have been rejected already in Game 1, and we get

$$\Pr[out_2 = 1] = \Pr[out_1 = 1]. \tag{38}$$

In **Game** 3, we slightly change how openings of ciphertexts are performed. In Game 2, we have revealed the random coins R_{t_p} used to construct the ABM core tag part t_p upon an opening. In Game 3, we instead reveal random coins that are generated via $R_{t_\mathsf{p}} \leftarrow \mathsf{Expl}_{\mathcal{T}_\mathsf{p}}(1^k, t_\mathsf{p})$. Since the core tag space $\mathcal{T}_\mathsf{p}$ is efficiently samplable and explainable by assumption, we get

$$\Pr[out_3 = 1] = \Pr[out_2 = 1]. \tag{39}$$

In **Game** 4, we generate all ABM tags in the $C^{(i)}$ as lossy tags, using $t_\mathsf{p} \leftarrow \mathcal{T}_{\mathsf{loss}}((\rho, Y'))$. A straightforward reduction shows

$$\Pr[out_4 = 1] - \Pr[out_3 = 1] = \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM},B}(k) \tag{40}$$

for a suitable adversary B on ABM's indistinguishability.

Recall that in the original scheme PKE, we first use LTF's inversion key ik' to obtain $X := f^{-1}_{ik'}(Y')$. We then check that $f_{ek,(t_\mathsf{p},(\rho,Y'))}(X) = Y$ (and reject if not). Now in **Game** 5, we use ABM's inversion key ik to first obtain $X := f^{-1}_{ek,(t_\mathsf{p},(\rho,Y'))}(Y)$ and then check that $f_{ek'}(X) = Y'$. By the correctness properties of LTF and ABM, these procedures yield the same results, *unless* the adversary submits a decryption query with a non-injective, non-copied tag. We thus need to bound $\Pr[\mathsf{bad}_{\mathsf{ninj}}]$, where $\mathsf{bad}_{\mathsf{ninj}}$ denotes the event that A submits a decryption query with a non-injective ABM tag $t = (t_\mathsf{p}, (\rho, Y'))$ that is not copied. However, the evasiveness property of ABM guarantees that

$$|\Pr[out_5] - \Pr[out_4]| \leq \Pr[\mathsf{bad}_{\mathsf{ninj}}] \leq q(k) \cdot \mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM},C}(k) \tag{41}$$

is negligible, where B is a suitable adversary against the evasiveness property of ABM. (Concretely, B simulates Game 4, chooses $i \in [q]$ uniformly, and outputs the tag $t = (t_\mathsf{p}, (\rho, Y'))$ from A's ith decryption query if it is not copied. Note that B can use its $\mathsf{ABM.LTag}$ oracle to produce lossy ABM tags.)

In **Game** 6, we generate LTF's evaluation key ek' as a lossy key, via $ek' \leftarrow \mathsf{LTF.LGen}(1^k)$. Since in Game 5, LTF's inversion key ik' is never used, a straightforward reduction shows

$$\Pr[out_5 = 1] - \Pr[out_6 = 1] = \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},D}(k) \tag{42}$$

for a suitable PPT adversary on LTF's indistinguishability.

We can now observe that in Game 6, all evaluations $Y' = f_{ek'}(X)$, resp. $Y = f_{ek,(t_\mathsf{p},(\rho,Y'))}(X))$ that A receives in the challenge ciphertexts are made with respect to lossy keys, resp. tags. In particular, at this point, the tags $h(X)$ generated during encryption to pad the actual message M are statistically close to uniform, *even given* Y' *and* Y. Since the message vectors $\mathbf{M}_0$ and $\mathbf{M}_1$ from $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\mathsf{PKE},A}$ are thus almost[6] identically distributed, we finally obtain

$$\Pr\left[out_6 = 1\right] = \frac{1}{2} + N \cdot \mathbf{O}(2^{-2k}) = \frac{1}{2} + \mathbf{O}(2^{-k}). \tag{43}$$

Taking (37-43) together shows (36). □

7 From indistinguishability-based to simulation-based security

Motivation. While our used definition of selective opening security (IND-SO-CCA security) is common, it has certain drawbacks. To explain, note that the IND-SO-CCA security experiment involves a conditional re-sampling of messages $\mathbf{M}_1 := \mathsf{ReSamp}_{\mathsf{dist}}(\mathbf{M}_\mathcal{I})$. This operation is only efficient if the message distribution is efficiently re-samplable. This causes two problems.

First, in settings that correspond to arbitrary message distributions, IND-SO-CCA security may lead to inefficient hybrid settings. For instance, consider a game-based proof that first uses the IND-SO-CCA security of an encryption scheme and thus substitutes encrypted messages with re-sampled ones. This leads to a game that re-samples messages and thus may itself not be efficient. However, once an intermediate game is inefficient, later reductions to a computational assumption may be considerably more difficult if not impossible.

Second, we can only *prove* IND-SO-CCA security with respect to efficiently re-samplable message distributions. The reason is essentially the same: once the re-sampling of messages is inefficient, we cannot even simulate the IND-SO-CCA security experiment efficiently. This hinders reductions to a computational assumption (e.g., the indistinguishability or evasiveness of an ABM-LTF).

Simulation-based selective opening security. Another notion of selective opening security is the simulation-based notion of SIM-SO-CCA security [5, 23, 18]. With this notion, we require that an SO-CCA attack on the scheme can be simulated in a setting in which only the opened messages (but no ciphertexts at all) are available to the simulator. SIM-SO-CCA security has the advantage that the security experiments (real and ideal) are efficient. In particular, it is possible to show security with respect to arbitrary message distributions [5, 23, 18].

We now briefly sketch how we can achieve SIM-SO-CCA security (with respect to arbitrary message distributions) for our DCR-based construction. To show SIM-SO-CCA security, we need to devise an *efficient* algorithm Opener that opens a given lossy ciphertext arbitrarily, i.e., as an encryption of an arbitrary message. Here, a lossy ciphertext denotes one which is encrypted with respect to a lossy LTF key, and a lossy ABM-LTF tag. (See [5] for details.)

SIM-SO-COM security of our DCR-based scheme. So consider the scheme PKE from Section 6.3, instantiated with our DCR-based ABM-LTF ABMD and the DCR-based LTF from [20]. For ease of presentation, assume that a preimage $X \in \mathbb{Z}^3_{N^s}$ is chosen as a preimage for both ABM-LTF and LTF. The message-dependent part of a ciphertext is then $\rho = h(X) \oplus M$ for a suitable hash function h yet to be described. With a lossy ABM-LTF tag and a lossy LTF key, the remaining ciphertext only depends on the following values:

$$\widetilde{M} \cdot X \bmod \mathbb{Z}^3_{N^s}, \qquad X \bmod \varphi(N), \qquad X \bmod \varphi(N'), \tag{44}$$

where $\widetilde{M}$ is the (rank-2) matrix from (2), and N' is the modulus employed by the DCR-based LTF from [20]. Now suppose we initially select h as the inner product $h(X) := V \cdot X$ for a uniform

[6] Recall that $\mathbf{M}_1$ is sampled from a distribution that is conditioned on a set $\mathcal{I}$ that itself may depend on $\mathbf{C}$. Since $\mathbf{C}$ may still contain negligible information about the encrypted messages $\mathbf{M}_0$, there may be a (negligible) statistical difference between the distributions of $\mathbf{M}_0$ and $\mathbf{M}_1$.

$V \in \mathbb{Z}_{N^s}^3$. Then, with overwhelming probability, V is linearly independent of the rows of the (rank-2) matrix $\widetilde{M}$. This means that, given $\widetilde{M}$, X, and V, we can efficiently find uniformly distributed X' that (a) are consistent with the values from (44), and (b) fulfill $h(X') \oplus \rho = M$ for any given ρ, M. This means that lossy ciphertexts can be efficiently opened as encryptions of arbitrary messages, and SIM-SO-CCA security follows.

The case of our pairing-based scheme. It would seem natural to expect that a similar technique can be used to prove the SIM-SO-CCA security of PKE, instantiated with our pairing-based ABM-LTF ABMP and, say, the DDH-based LTF from [32]. To explain the problem that arises, note that preimages X then are from $\{0,1\}^n$, and a ciphertext fixes

$$\sum_{i=1}^{n} \alpha_i X_i \bmod p \qquad \text{and} \qquad \sum_{i=1}^{n} \alpha'_i X_i \bmod p', \tag{45}$$

where α_i, α'_i are exponents induced by the ABM-LTF tag, resp. the LTF key, and p, p' are corresponding group orders. We could choose a suitable hash function of the form $h(X) = \sum_{i=1}^{n} \beta_i X_i$ with uniform β_i. (It can be expected that the vector $(\beta_i)_i$ is linearly independent of the vectors $(\alpha_i)_i$ and $(\alpha'_i)_i$.) However, to construct an efficient Opener algorithm, we would have to come up with $X' \in \{0,1\}^n$ that are consistent with (45) and satisfy $h(X') \oplus \rho = M$ for given ρ, M. By linearity, this would require finding uniformly distributed *binary* solutions of the equations

$$\sum_{i=1}^{n} \alpha_i X_i = 0 \bmod p \qquad \text{and} \qquad \sum_{i=1}^{n} \alpha'_i X_i = 0 \bmod p'.$$

This seems to require at least a special setup of the α_i, α'_i, which in turn is not consistent with our computational assumptions.

Acknowledgements

The author would like to thank Florian Böhl, Serge Fehr, Eike Kiltz, and Hoeteck Wee for helpful and insightful discussions concerning SO-CCA security. Furthermore, the anonymous Crypto 2011 referees have given useful comments that helped to improve the paper.

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] 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, Bruges, Belgium, May 14–18, 2000. Springer, Berlin, Germany.

[4] 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, Tokyo, Japan, December 6–10, 2009. Springer, Berlin, Germany.

[5] 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, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany.

[6] Mihir Bellare, Brent Waters, and Scott Yilek. Identity-based encryption secure against selective opening attack. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 235–252, Providence, RI, USA, March 28–30, 2011. Springer, Berlin, Germany.

[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, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Berlin, Germany.

[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, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 325–341, Cambridge, MA, USA, February 10–12, 2005. Springer, Berlin, Germany.

[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, Beijing, China, June 22–25, 2010. Springer, Berlin, Germany.

[11] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648, Philadephia, Pennsylvania, USA, May 22–24, 1996. ACM Press.

[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, Interlaken, Switzerland, May 2–6, 2004. Springer, Berlin, Germany.

[13] 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, Cheju Island, South Korea, February 13–15, 2001. Springer, Berlin, Germany.

[14] 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, Santa Barbara, CA, USA, August 20–24, 2000. Springer, Berlin, Germany.

[15] 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, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Berlin, Germany.

[16] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *23rd ACM STOC*, pages 542–552, New Orleans, Louisiana, USA, May 6–8, 1991. ACM Press.

[17] Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. Magic functions. In *40th FOCS*, pages 523–534, New York, New York, USA, October 17–19, 1999. IEEE Computer Society Press.

[18] 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, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[19] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 44–61, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[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, Paris, France, May 26–28, 2010. Springer, Berlin, Germany.

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

[25] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS 2000*, San Diego, California, USA, February 2–4, 2000. The Internet Society.

[26] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479, Zurich, Switzerland, February 9–11, 2010. Springer, Berlin, Germany.

[27] Sarah Meiklejohn, Hovav Shacham, and David Mandell Freeman. Limitations on transformations from composite-order to prime-order groups: The case of round-optimal blind signatures. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 519–538, Singapore, December 5–9, 2010. Springer, Berlin, Germany.

[28] Sarah Meiklejohn, Hovav Shacham, and David Mandell Freeman. Limitations on transformations from composite-order to prime-order groups: The case of round-optimal blind signatures. In *ASIACRYPT*, pages 519–538, 2010.

[29] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press.

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

[31] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.

[32] 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, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press.

[33] 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, Berlin, Germany, March 15–17, 2009.

[34] Victor Shoup. On formal models for secure key exchange. Technical Report RZ 3120, IBM, 1999.

[35] Markus Stadler. Publicly verifiable secret sharing. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 190–199, Saragossa, Spain, May 12–16, 1996. Springer, Berlin, Germany.

[36] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636, Santa Barbara, CA, USA, August 16–20, 2009. Springer, Berlin, Germany.

[37] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.