New Techniques for Efficient Trapdoor Functions and Applications

Sanjam Garg[1], Romain Gay[2,3], and Mohammad Hajiabadi[1,4]

[1] University of California, Berkeley
{sanjamg,mdhajiabadi}@berkeley.edu
[2] Département informatique de l'ENS, École normale supérieure, CNRS, PSL University, 75005 Paris, France
romain.gay@ens.fr
[3] INRIA, Paris, France
[4] University of Virginia.

Abstract. We develop techniques for constructing trapdoor functions (TDFs) with short image size and advanced security properties. Our approach builds on the recent framework of Garg and Hajiabadi [CRYPTO 2018]. As applications of our techniques, we obtain

- The first construction of deterministic-encryption schemes for block-source inputs (both for the CPA and CCA cases) based on the Computational Diffie-Hellman (CDH) assumption. Moreover, by applying our efficiency-enhancing techniques, we obtain CDH-based schemes with ciphertext size linear in plaintext size.
- The first construction of lossy TDFs based on the Decisional Diffie-Hellman (DDH) assumption with image size linear in input size, while retaining the lossiness rate of [Peikert-Waters STOC 2008].

Prior to our work, all constructions of deterministic encryption based even on the stronger DDH assumption incurred a quadratic gap between the ciphertext and plaintext sizes. Moreover, all DDH-based constructions of lossy TDFs had image size quadratic in the input size.

At a high level, we break the previous quadratic barriers by introducing a novel technique for encoding input bits via hardcore output bits with the use of erasure-resilient codes. All previous schemes used group elements for encoding input bits, resulting in quadratic expansions.

1 Introduction

Trapdoor functions (TDFs) are a fundamdental primitive in cryptography and are typically used as a fundamental building block in the construction of advanced primitives such as CCA2-secure public-key encryption (PKE). Introduced in the 70's [DH76, RSA78], TDFs are a family of functions, where each individual function in the family is easy to compute, and also easy to invert if one posses an additional trapdoor key. The basic security requirement is that of one-wayness, requiring that a randomly chosen function from the family be one-way.

The usefulness of TDFs stems from the fact that the inversion algorithm recovers the entire input. This stands in sharp contrast to PKE, wherein the

decryption algorithm may not recover the underlying randomness. This input recovery feature of TDFs is what makes them a useful tool, especially in applications where proofs of well-formedness are required.

On the other hand, building TDFs turns out to be much more difficult than building PKE, mostly due to the requirement of recovering the entire input, which in turn is the reason behind the lack of black-box transformations from PKE to TDFs [GMR01]. Specifically, in groups with discrete-log based hardness assumptions, this restricts the use of operations such as *exponentiation*, for which we do not have a generic trapdoor. Furthermore, in some applications we need TDFs to be *robust*, providing enhanced security properties rather than mere one-wayness (e.g., [BBO07, PW08, PW11, PW11, PVW08, BFOR08, RS09]).

Recently, Garg and Hajiabadi [GH18] introduced a new approach for building TDFs, obtaining the first construction of TDFs from the Computational Diffie-Hellman (CDH) assumption. Although their approach gives new feasibility results, their constructed TDFs are limited in certain ways: (a) Their TDFs are not robust enough — for example, it is not clear how to go beyond one-wayness, obtaining more advanced properties such as those required by deterministic encryption [BBO07, BFOR08, BFO08] or CCA2 security; and (b) The length of their TDF images grows (at least) quadratically with the length of the input.

We stress that Point (b) is not just an artifact of the construction of [GH18]. In fact, we do not know of any TDF constructions (even based on the stronger decisional Diffie-Hellman (DDH) assumption) with advanced properties, such as deterministic-encryption security, with images growing linearly in their inputs.[5] Since TDFs are typically used as building blocks in more advanced primitives, designing more efficient TDFs translates into the same features in target applications. For example, lossy TDFs [PW08, PW11, PW11] are an extremely versatile primitive with a long list of applications; e.g., [BFOR08, BHY09, BBN+09, MY10, BCPT13].

1.1 Our Results

We develop techniques for constructing efficient and robust TDFs. As concrete applications of our new techniques, we obtain the first construction of deterministic encryption for block sources (in the sense of [BFO08]) under the CDH assumption. We give both CPA and CCA2 versions of our constructions. We stress that prior to our work we knew how to build (even) CPA-secure deterministic encryption only from decisional assumptions, including DDH, QR, DCR and LWE [BFO08, Wee12]. Thus, in addition, we also obtain instantiations under the hardness of factoring assumption.

Furthermore, we show how to use our efficiency techniques to obtain:

[5] We note that building a TDF providing mere one-wayness with linear-size images is simple: if $\mathsf{TDF.F}(\mathsf{ik}, \cdot)$ maps n-bit inputs to n^c-bit outputs, define $\mathsf{TDF.F'}(\mathsf{ik}, \mathsf{x}||\mathsf{x}')$, where $|\mathsf{x}| = n$ and $|\mathsf{x}'| = n^c$, as $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})||\mathsf{x}'$. Although this transformation results in TDFs with linear-image size, it destroy more advanced properties such as CCA2 security, deterministic-encryption security and the lossiness rate.

1. The first CDH-based deterministic encryption schemes with ciphertext size linear in plaintext size. Additionally, our CDH-based deterministic-encryption schemes beat all the previous DDH-based schemes in terms of ciphertext size. The sizes of other parameters (e.g., the secret key and public key) remain the same. See Table 1 for a comparison.
2. The first construction of lossy TDFs ([PW08, PW11, PW11]) from DDH with image size linear in input size. Our DDH-based lossy TDFs achieve the same lossiness rate as in [PW08, PW11, PW11]. All previous DDH-based lossy TDF constructions (achieving non-trivial lossiness rates) resulted in images quadratically large in their inputs.

work	assumption	primitive	index key	trapdoor key	image
ours	CDH	CCA2 DE	$\Theta(n^2 \log p)$	$\Theta(n^2 \log p)$	$\log p + \Theta(n)$
[**BFO08**]	DDH	CCA2 DE	$\Theta(n^2 \log p)$	$\Theta(n^2 \log p)$	$\Theta(n \log p)$
ours	DDH	$(n, \log p)$-LTDF	$\Theta(n^2 \log p)$	$\Theta(n^2 \log p)$	$\log p + \Theta(n)$
[**PW08, PW11, PW11, FGK+10**]	DDH	$(n, \log p)$-LTDF	$\Theta(n^2 \log p)$	$\Theta(n^2 \log p)$	$\Theta(n \log p)$

Table 1: Bit complexity: p is the order of the group and n is the bit size of the TDF input. Here (n, k)-LTDF means lossy TDFs where in lossy mode the image-space size is at most 2^k. We call $1 - k/n$ the lossiness rate.

1.2 Technical Overview

In this section we give an overview of our techniques for constructing robust and efficient TDFs. We will build TDFs with several abstract properties, and we will apply these techniques to the setting of deterministic encryption and lossy TDFs as concrete applications.

Our constructions rely on the same primitive of *recyclable one-way function with encryption (OWFE)* used by [GH18], so we first review this notion. An OWFE consists of a one-way function $\mathsf{f}(\mathsf{pp}, \cdot)\colon \{0,1\}^n \to \{0,1\}^\nu$, where pp is a public parameter, along with encapsulation/decapsulation algorithms (E, D). Specifically, E takes as input pp, an image $\mathsf{y} \in \{0,1\}^\nu$ of $\mathsf{f}(\mathsf{pp}, \cdot)$, a target index $i \in [n]$ and a target bit $b \in \{0,1\}$, and produces an *encapsulated ciphertext* ct and a corresponding *key bit* $\mathsf{e} \in \{0,1\}$. The algorithm D allows us to retrieve e from ct using any pre-image x of y, if $\mathsf{x}_i = b$. For security, letting $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$, we require that if $(\mathsf{ct}, \mathsf{e}) \xleftarrow{\$} \mathsf{E}(\mathsf{pp}, \mathsf{y}, (i, 1 - \mathsf{x}_i))$, then even knowing both x and ct, one cannot distinguish e from a truly random bit. Finally, letting E_1 and E_2 refer to the first and second output pars of E, the recyclability requirement says

that the output of E_1 does not depend on y, namely, we have: $\mathsf{ct} = \mathsf{E}_1(\mathsf{pp}, (i, b))$ and $\mathsf{e} = \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, b))$. (See Definition 4.) The work of [GH18] gives CDH instantiations of this notion; see Section **??**.

Approach of [GH18]. A property implied by recyclable OWFE is the following: given $\mathsf{x} \in \{0,1\}^n$ and two fresh encapsulated ciphertexts $(\mathsf{ct}_0, \mathsf{ct}_1)$ made w.r.t. $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$ and an arbitrary target index i and target bits 0 and 1 (respectively), one cannot distinguish the values of the corresponding two key bits $(\mathsf{e}_0, \mathsf{e}_1)$ from a pair in which we replace $\mathsf{e}_{1-\mathsf{x}_i}$ with a random bit. Exploiting this property, [GH18] set their index key to contain encapsulated ciphertexts $\mathsf{ct}_{i,b}$ made w.r.t. each value of $i \in [n]$ and $b \in \{0,1\}$ — they put all the corresponding randomness values $[r_{i,b}]$ in the trapdoor key. The input to their TDF contains $\mathsf{x} \in \{0,1\}^n$ and the output u consists of $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$ as well as a $2 \times n$ matrix $\mathbf{M}$ of bits $(\mathsf{e}_{i,b})_{i \in [n], b \in \{0,1\}}$, where for all i, they set $\mathsf{e}_{i,\mathsf{x}_i} := \mathsf{D}(\mathsf{pp}, \mathsf{x}, \mathsf{ct}_{i,\mathsf{x}_i})$ and set $\mathsf{e}_{i,1-\mathsf{x}_i}$ to be a random bit. Since TDFs are not allowed to make use of randomness, they draw $\mathsf{e}_{i,1-\mathsf{x}_i}$ for all i from an additional part of their input which they call the *blinding* part. For inverting $\mathsf{u} := (\mathsf{y}, \mathbf{M})$, the inverter may make use of its knowledge of all the randomness values underlying $\mathsf{ct}_{i,b}$'s to form the corresponding key bits w.r.t. y. Then the inverter may check each column of the resulting matrix, $\mathbf{M}'$, against the corresponding column of the matrix $\mathbf{M}$, and look for a matched coordinate. This would enable recovering half of the input bits (on average). The one-wayness of their scheme follows by the property alluded to above. Namely, for any $i \in [n]$, we may switch $\mathsf{e}_{1-\mathsf{x}_i}$ from uniformly random to $\mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, 1-\mathsf{x}_i); r_{i,1-\mathsf{x}_i})$. Consequently, the image of the trapdoor function becomes: $(\mathsf{y}, (\mathsf{e}_{i,b})_{i,b})$, where $\mathsf{e}_{i,b} := \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, b); r_{i,b})$ for all $i \in [n]$ and $b \in \{0,1\}$. In other words, the entire view of a TDF adversary may be computed from y alone. At this point, the one-wayness of the TDF follows from the one-wayness of the underlying OWFE. Finally, [GH18] boosts correctness by repeating the above process in parallel. For future reference, we call the above initial TDF (which enables the recovery of half of the bits) *TDF gadget*.

Lack of perfect correctness in [GH18]. The TDF of [GH18] only achieves a weak form of correctness, under which the inversion algorithm may fail w.r.t. *any* index/trapdoor keys for a negligible fraction of the inputs. This severely restricts the applicability of CCA2-enhancing techniques, such as those of [RS09, KMO10], for obtaining CCA2 secure primitives. Even for the CPA case, the lack of perfect correctness hindered the construction of CPA-secure deterministic encryption schemes. Deterministic public-key encryption schemes [BBO07] are TDFs which hide information about plaintexts drawn from high min-entropy sources. There are various forms of this definition, e.g., [BBO07, BFO08, BFOR08, BS11, MPRS12]. Strong versions of this notion have so far been realized in the random oracle model [BBO07] and are subject to impossibility results [Wic13]. Boldyreva, Fehr and O'Neill [BFO08] formulated a relaxation of this notion (called *block-source security*), and showed how to realize this relaxed notion under standard assumptions such as DDH and LWE. Informally, block-source security requires that the (deterministic) encryptions of any two sources with

high min entropy (more than a threshold k) remain computationally indistinguishable.[6] Ideally, we want $k << n$, where n is plaintext size.

The TDF of [GH18] does not achieve block-source security for the same reason that degraded their correctness property: The TDF input contains a blinding part, which in turn is copied in the clear in the output (but in hidden spots). To see how this breaks security, consider two sources, where the first one fixes the blinding part to all zeros, and the second one fixes it to all ones. Then it would be easy to distinguish between the outputs of the TDF w.r.t. these two sources, even though they may have high min entropy.

Enhancing to perfect correctness. We fix the imperfect correctness of [GH18] via a *mirroring* technique. Recall that the bits of the blinding input were previously used to form the values of $\mathsf{e}_{i,1-\mathsf{x}_i}$. Now, instead of having a blinding part in the input for making up the values of $\mathsf{e}_{i,1-\mathsf{x}_i}$, we set $\mathsf{e}_{i,1-\mathsf{x}_i} := \mathsf{a}_i - \mathsf{e}_{i,\mathsf{x}_i}$, where $\mathbf{a} := (\mathsf{a}_1, \ldots, \mathsf{a}_n) \in \{0,1\}^n$ is a random vector that comes from the index key. This way we get rid of inclusion of blinders as part of the input — the input now solely consists of a string $\mathsf{x} \in \{0,1\}^n$. We show that this method improves correctness: Our TDF is now perfectly correct for all but a negligible fraction of index/trapdoor keys; see Remark 1.

Lossy-like properties of our TDF toward obtaining deterministic encryption. So far, we showed how to fix the imperfect-correctness problem of [GH18], but this by itself does not guarantee deterministic-encryption security. Toward this goal, we show that the mirroring technique allows us to establish a *lossy-like* property for our TDFs, which in turn gives us block-source security.[7] Specifically, let y be an image point of $\mathsf{f}(\mathsf{pp}, \cdot)$ of the OWFE scheme, and let S be the set of all pre-images of y (which can be of exponential size under our CDH instantiation). We can now set the index key as ik_y, where (a) ik_y loses information w.r.t. all pre-images of y: for all $\mathsf{x}, \mathsf{x}' \in \mathsf{S}$ we have $\mathsf{TDF.F}(\mathsf{ik}_\mathsf{y}, \mathsf{x}) = \mathsf{TDF.F}(\mathsf{ik}_\mathsf{y}, \mathsf{x}')$ and (b) ik_y is computationally indistinguishable from an honestly generated ik. We exploit this property to prove block-source security for our TDFs.

Having achieved block-source CPA security, we may boost this scheme into a CCA2-secure deterministic-encryption scheme using the techniques of [RS09, KMO10].[8] Specifically, we show how to use our lossiness property to prove k-*repetition* security (introduced by [RS09]) for our TDF. Intuitively, k-repetition security requires one-wayness to hold even if the given input is evaluated under

[6] This is the indistinguishability-based, single-message version of their notion, which as they show, is equivalent to the multiple-message version both for the indistinguishability- and simulation-based definitions.

[7] We note that this lossiness property is weaker than the one of [PW08, PW11, PW11], but it can be realized under CDH. We will later show efficient DDH-based instantiations of lossiness in the sense of [PW08, PW11, PW11].

[8] We mention that the transformation of [RS09] results in CCA-secure PKE schemes which use randomness, but this can be avoided by using the techniques of [KMO10] to get CCA2-secure TDFs.

k-randomly chosen functions. The scheme of [GH18] fails to achieve k-repetition security, exactly because of the presence of blinders.

Finally, we mention that based on CDH we do not get lossiness in the sense of [PW08, PW11] as the amount of information we lose is negligible over the entire input space. Nevertheless, our weak lossiness property which can be realized under CDH suffices for our deterministic-encryption application, and may find other applications later.

Efficiency of our TDFs so far: quadratically large images. Under CDH instantiations of the above approach, for plaintexts of n bits, the bit-size of the ciphertext is $\Theta(n^2)$ in the CPA case, and $\Theta(n^2\omega(\log n))$ in the CCA case. In contrast, the DDH-based constructions of [BFO08] give ciphertext size $\Theta(n^2)$ both for the CPA and CCA cases.

Sources of inefficiency. Recall our TDF gadget has image size $\Theta(n)$. This TDF gadget may fail to recover any given bit of the input with probability 1/2. Thus, we ran many TDF gadgets in parallel, resulting in $\Theta(n^2)$ image size. We refer to this as *correctness repetition*. For the CCA2 case, since we relied on techniques of [RS09, KMO10] we needed to perform yet another repetition, which we call *CCA2 repetition*. This justifies the further blowup in CCA2 image size.

We develop techniques for avoiding both these repetitions, sketched below.

Erasure-resilient codes to the rescue: linear-image TDFs. We give techniques involving the use of erasure-resilient codes for making the size of our TDF images linear, while preserving other properties. Recall that under our TDF gadget, for a randomly chosen input $\mathsf{x} \in \{0,1\}^n$ and for any index $i \in [n]$, the inversion algorithm either recovers x_i correctly, or outputs $\perp$ for this bit position (with probability 1/2). Notice that the inversion process has a *local property*, in the sense that each bit individually may be recovered or not with probability 1/2.

Now instead of performing parallel repetition which results in a quadratic-size blowup, we boost correctness through the use of erasure-resilient codes. Suppose $(\mathsf{Encode}, \mathsf{Decode})$ is an erasure-resilient code, where $\mathsf{Encode}\colon \{0,1\}^n \to \{0,1\}^m$ (for $m = cn \in O(n)$), and where Decode only needs $c_1 n$ (noise-free) bits of a codeword $\mathsf{Encode}(\mathsf{x})$ — for c_1 sufficiently smaller than c — in order to recover x. Such codes may be built from Reed-Solomon codes by adapting them to bit strings; see Definition 8.

The starting point of our technique is the following: On input $\mathsf{x} \in \{0,1\}^n$, apply the TDF gadget on the encoded input $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$. To invert, we no longer need to recover *all* the m bits of z; recovering $c_1 n$ of them will do. Unfortunately, for codes over binary alphabets, the value of c_1/c is much greater than 1/2 and our TDF gadget may be incapable of recovering that many bits. We get around this issue by doing repetition but for a constant number of times: Instead of applying the TDF gadget to $\mathsf{z} := \mathsf{z}_1 \ldots \mathsf{z}_m$, apply it to the t-repetition copy of z where we repeat each bit of z t times. By choosing the underlying constants appropriately and using the mirroring idea, we can ensure perfect correctness for all but a negligible fraction of index/trapdoor keys. This way,

images will grow linearly. The proof of CPA block-source security follows almost similarly as before.

Concretely, under CDH instantiations, CPA ciphertexts (for plaintexts of n bits) consist of one group element and a constant number of bits.[9] This substantially improves the ciphertext size of previous DDH-based schemes under which a ciphertext consists of n group elements.

Keeping image size linear in CCA-like applications. So far, we showed how to build linear-image TDFs with additional properties (e.g., block-source CPA security). Typically, TDFs with enhanced properties (such as k-repetition security [RS09] or lossy properties [PW08, PW11]) can be boosted into CCA2 primitives, but this requires "parallel repetition" of the base object, increasing the sizes. Our linear-image TDF turns out to be k-repetition secure, but we cannot afford to use previous techniques for getting CCA2 security, because we would like to keep image size linear. Here is where our other techniques come into play: We develop a half-simulation method for proving CCA2 security for our *same* TDF scheme without any further modifications. For this, we just need to choose the constant c in $m = cn$ big enough. Our CCA techniques are different from those of [PW08, PW11, RS09], which implicitly or explicitly relied on repetition.

As an application, we will get a CDH-based block-source CCA-secure deterministic encryption, beating the ciphertext size of DDH-based schemes. We now sketch our techniques.

Let $(\mathsf{Encode}, \mathsf{Decode})$ be a code obtained by repeating the output bit of a codeword (which in turn is obtained based on a linear-size-output error-correcting code) t times for a constant t. See Definition 8 and the two paragraphs afterward. A codeword z may be thought of as a string of m/t blocks, each consisting of entirely either t zeros or t ones.

Recall that a trapdoor key tk consists of all randomness values $r_{i,b}$'s used to form $\mathsf{ct}_{i,b}$'s (which are in turn fixed in the index key ik). On input $\mathsf{x} \in \{0,1\}^n$ we form $\mathsf{z} := \mathsf{Encode}(\mathsf{x}) \in \{0,1\}^m$ and return $\mathsf{u} := (\mathsf{y}, \mathbf{M} := \left(\begin{smallmatrix} \mathsf{e}_{1,0},\ldots,\mathsf{e}_{m,0} \\ \mathsf{e}_{1,1},\ldots,\mathsf{e}_{m,1} \end{smallmatrix}\right))$, where $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{z})$, $\mathsf{e}_{i,\mathsf{z}_i} = \mathsf{D}(\mathsf{pp}, \mathsf{z}, \mathsf{ct}_{i,\mathsf{z}_i})$ and $\mathsf{e}_{i,1-\mathsf{z}_i} = \mathsf{a}_i - \mathsf{e}_{i,\mathsf{z}_i}$, where $\mathbf{a} := (\mathsf{a}_1, \ldots, \mathsf{a}_m) \in \{0,1\}^m$ is sampled in ik. The inversion algorithm will recover the ith bit of z iff $\mathsf{e}_{i,1-\mathsf{z}_i} = 1 - \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, 1-\mathsf{z}_i); r_{i,1-\mathsf{z}_i})$. Say the ith column of $\mathbf{M}$ is *hung* if $\mathsf{e}_{i,1-\mathsf{z}_i} = \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, 1-\mathsf{z}_i); r_{i,1-\mathsf{z}_i})$ — if this happens, then the inverter cannot decide on the ith bit of z.

Let us argue CCA2 security w.r.t. two sources $\mathcal{S}_0$ and $\mathcal{S}_1$: The adversary should distinguish $(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}_0))$ from $(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}_1))$, where $\mathsf{x}_b \xleftarrow{\$} \mathcal{S}_b$. For deterministic encryption we may assume all CCA queries happen after seeing the index key and challenge ciphertext (Definition 2).

Our CCA2 simulation is based on a *half-trapdoor simulation* technique under which we forget one randomness value from each pair $(r_{i,0}, r_{i,1})$ in the trapdoor. Specifically, letting $\mathsf{x}^\star$ be the challenge plaintext, imagine a half-trapdoor key obtained based on $\mathsf{x}^\star$ from tk as $\mathsf{tk}_{\mathrm{rd},\mathsf{z}^\star} := (r_{1,\mathsf{z}^\star_1}, \ldots, r_{m,\mathsf{z}^\star_m})$, where $\mathsf{z}^\star = \mathsf{Encode}(\mathsf{x}^\star)$.

[9] We have not yet optimized nor tried to get some upper bounds on the constants.

We perform half-trapdoor inversion of a given point $\mathsf{u} := \left(\mathsf{y}, \left(\begin{smallmatrix}\mathsf{e}_{1,0},\ldots,\mathsf{e}_{m,0}\\ \mathsf{e}_{1,1},\ldots,\mathsf{e}_{m,1}\end{smallmatrix}\right)\right)$ w.r.t. $\mathsf{tk}_{\mathrm{rd},\mathsf{z}^\star}$ as follows: Build (potentially) a noisy codeword z as follows: in order to recover the bits of the jth block, if at least for *one* index i in this block we have $\mathsf{e}_{i,\mathsf{z}_i^\star} = 1 - \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, \mathsf{z}_i^\star); r_{i,\mathsf{z}_i^\star})$, set all the bits of z in this block to $1 - \mathsf{z}_i^\star$; otherwise, set all those bits to the corresponding bit values of $\mathsf{z}^\star$ in those coordinates. Once z is formed, decode it to get a string and check if the re-encryption of that string gives back u. If so, return the string.

Letting $\mathsf{x}^\star$ be the challenge plaintext (and recalling that all CCA queries are post-challenge), we first show we may use $\mathsf{tk}_{\mathrm{rd},\mathsf{z}^\star}$ (instead of the full key tk) to reply to the CCA queries, without the adversary noticing any difference, using the following two facts. First, for a queried point u, if u is not a valid image (i.e., it does not have a pre-image), then both (full and half) inversions return $\perp$. This is because at the end of either inversion we re-encrypt the result to see whether we get the given image point back. So suppose for the queried $\mathsf{u} := \left(\mathsf{y}, \mathbf{M} := \left(\begin{smallmatrix}\mathsf{e}_{1,0},\ldots,\mathsf{e}_{m,0}\\ \mathsf{e}_{1,1},\ldots,\mathsf{e}_{m,1}\end{smallmatrix}\right)\right)$ we have $\mathsf{u} := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$ for some $\mathsf{x} \in \{0,1\}^n \setminus \{\mathsf{x}^\star\}$. (If $\mathsf{x} = \mathsf{x}^\star$, then u will be the challenge ciphertext itself and hence not a permitted query.) Let $\mathsf{S} \subseteq [m/t]$ contain the indices of those blocks on which z and $\mathsf{z}^\star$ differ, where $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$. Note the half-trapdoor inversion w.r.t. $\mathsf{tk}_{\mathrm{rd},\mathsf{z}^\star}$ will correctly recover *all* the bits of z that correspond to the blocks which are not in S.

For the blocks in S, we show that by choosing the constants appropriately, then for sufficiently-many indices $j \in \mathsf{S}$, the jth block of $\mathbf{M}$ is *not* hung; namely, for at least one index i in this block we have $\mathsf{e}_{i,1-\mathsf{z}_i} = 1 - \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, 1 - \mathsf{z}_i); r_{i,1-\mathsf{z}_i})$. For any index $j \in \mathsf{S}$ such that the above holds, the half-inversion process (w.r.t. $\mathsf{tk}_{\mathrm{rd},\mathsf{z}^\star}$) will recover the jth block of z (by definition). We will use these facts to argue we will have enough correctly generated bits in order to able to do error correction.

Once we solely use $\mathsf{tk}_{\mathrm{rd},\mathsf{z}^\star}$ to reply to decryption queries, letting $\mathsf{u}^\star := \left(\mathsf{y}^\star, \left(\begin{smallmatrix}\mathsf{e}_{1,0}^\star,\ldots,\mathsf{e}_{m,0}^\star\\ \mathsf{e}_{1,1}^\star,\ldots,\mathsf{e}_{m,1}^\star\end{smallmatrix}\right)\right)$ be the corresponding challenge ciphertext, we may replace each $\mathsf{e}_{i,1-\mathsf{z}_i^\star}^\star$ with $\mathsf{E}_2(\mathsf{pp}, \mathsf{y}^\star, (i, 1 - \mathsf{z}_i^\star); r_{i,1-\mathsf{z}_i^\star})$, and simultaneously set the ith bit of the vector $\mathbf{a}$ of the index key as $\mathsf{a}_i := \mathsf{E}_2(\mathsf{pp}, \mathsf{y}^\star, (i, \mathsf{z}_i^\star); r_{i,\mathsf{z}_i^\star}) + \mathsf{E}_2(\mathsf{pp}, \mathsf{y}^\star, (i, 1 - \mathsf{z}_i^\star); r_{i,1-\mathsf{z}_i^\star})$. This change goes unnoticed by the security of the OWFE. At this point the challenge ciphertext and index key only depend on $\mathsf{y}^\star$ and we only use $\mathsf{z}^\star$ to decide which randomness value from each pair of tk to forget. We will now switch back to using the full trapdoor, with analysis similar to before. At this point, the entire view of the adversary may be simulated using $\mathsf{y}^\star := \mathsf{f}(\mathsf{pp}, \mathsf{z}^\star)$, and thus we have block-source security in this hybrid similar to the CPA case.

Lossy TDFs. Recall that a TDF is lossy [PW08, PW11] if one may generate index keys in a lossy way which is (1) indistinguishable from honestly generated index keys and (2) which results in statistical loss of information if used during the evaluation algorithm. We show we can adapt the trapdoor functions of [PW08, PW11] using our erasure-resilient code based technique for encoding input bits via hardcore output bits. This allows us to obtain lossy TDFs based on DDH with image size linear in input size. All previous DDH-based constructions of lossy TDFs incur a quadratic blowup in image size [PW08, PW11, FGK+10, Wee12].

We defer the reader to Section 6 for details. We leave open the exciting problem of constructing lossy trapdoor functions from CDH.

Other related work. OWFE is a relaxation of the notion of (chameleon) hash encryption and its variants, which in turn imply strong primitives such as laconic oblivious transfer and identity-based encryption (IBE) in a non-black-box way [CDG+17, DG17b, DG17a, BLSV18, DGHM18].

Freeman et al. [FGK+10] give additional constructions and simplifications to the TDF construction of [PW08, PW11]. Further constructions of (lossy) TDFs from various assumptions are given in [Wee12, HO12, HO13]. As for efficient TDFs, Boyen and Waters show that in the bilinear setting one may drastically shorten the index-key size of the Peikert-Waters lossy-TDF construction from a quadratic number of group elements to linear [BW10].

Concurrent Work. In an exciting independent and concurrent work, Koppula and Waters [KW18] show that TDF techniques can be used to upgrade any attribute-based encryption or predicate encryption scheme to its CCA secure variant. Similarly to this work, Koppula and Waters build on the ideas from the CDH-based TDF construction of Garg and Hajiabadi [GH18]. In particular, Koppula and Waters [KW18] independently came up with a similar version of the mirroring technique along the way, which we also developed in this paper. However, the focus of our work is very different from that of Koppula and Waters. In particular, we develop efficient techniques for applications such as TDFs, deterministic encryption and lossy trapdoor functions.

Paper organization. We give standard definitions and lemmas in Section 2 and OWFE-related definitions in Section 3. We give our (inefficient) construction of TDFs with deterministic-encryption security in Section 4 and give our efficient construction in Section 5. Finally, we give our DDH-based lossy TDF construction with linear image size in Section 6.

2 Preliminaries

Notation. We use λ for the security parameter. We use $\overset{c}{\equiv}$ to denote computational indistinguishability between two distributions and use $\equiv$ to denote two distributions are identical. For any $\varepsilon > 0$, we write $\approx_\varepsilon$ to denote that two distributions are statistically close, within statistical distance ε, and use $\overset{s}{\equiv}$ for statistical indistinguishability. For a distribution $\mathcal{S}$ we use $x \xleftarrow{\$} \mathcal{S}$ to mean x is sampled according to $\mathcal{S}$ and use $y \in \mathcal{S}$ to mean $y \in \sup(\mathcal{S})$, where sup denotes the support of a distribution. For a set S we overload the notation to use $x \xleftarrow{\$} \mathsf{S}$ to indicate that x is chosen uniformly at random from S. If $\mathsf{A}(x_1, \ldots, x_n)$ is a randomized algorithm, then $\mathsf{A}(a_1, \ldots, a_n)$, for deterministic inputs $a_1, \ldots, a_n$, denotes the random variable obtained by sampling random coins r uniformly at random and returning $\mathsf{A}(a_1, \ldots, a_n; r)$.

The min-entropy of a distribution $\mathcal{S}$ is defined as $\mathsf{H}_\infty(\mathcal{S}) \triangleq -\log(\max_{\mathsf{x}} \Pr[\mathcal{S} = \mathsf{x}])$. We call a distribution $\mathcal{S}$ a (k, n)-source if $\mathsf{H}_\infty(\mathcal{S}) \geq k$ and $\sup(\mathcal{S}) \subseteq \{0,1\}^n$.

2.1 Standard Definitions

Definition 1 (Trapdoor functions (TDFs)). *Let $n = n(\lambda)$ be a polynomial. A family of trapdoor functions* TDF *with domain $\{0,1\}^n$ consists of three PPT algorithms* $\mathsf{TDF.KG}$, $\mathsf{TDF.F}$ *and* $\mathsf{TDF.F}^{-1}$ *with the following syntax and security properties.*

- $\mathsf{TDF.KG}(1^\lambda)$*: Takes the security parameter 1^λ and outputs a pair $(\mathsf{ik}, \mathsf{tk})$ of index/trapdoor keys.*
- $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$*: Takes an index key ik and a domain element $\mathsf{x} \in \{0,1\}^n$ and deterministically outputs an image element u.*
- $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u})$*: Takes a trapdoor key tk and an image element u and outputs a value $\mathsf{x} \in \{0,1\}^n \cup \{\perp\}$.*

  We require the following properties.

- ***Correctness:***

$$\Pr_{(\mathsf{ik},\mathsf{tk})}[\exists \mathsf{x} \in \{0,1\}^n \text{ s.t. } \mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})) \neq \mathsf{x}] = \mathsf{negl}(\lambda), \quad (1)$$

  where the probability is taken over $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$.
- ***One-wayness:*** *For any PPT adversary $\mathcal{A}$, we have $\Pr[\mathcal{A}(\mathsf{ik}, \mathsf{u}) = \mathsf{x}] = \mathsf{negl}(\lambda)$, where $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$, $\mathsf{x} \xleftarrow{\$} \{0,1\}^n$ and $\mathsf{u} := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$.*

Remark 1. *The work of Garg and Hajiabadi [GH18] builds a TDF with a weaker correctness guarantee, under which for any choice of $(\mathsf{ik}, \mathsf{tk})$, we are allowed to have a negligible inversion error (over the choice of $\mathsf{x} \xleftarrow{\$} \{0,1\}^n$). Although the correctness condition of [GH18] implies that for a randomly chosen $(\mathsf{ik}, \mathsf{tk})$ and a randomly chosen x, the probability of an inversion error is negligible, it falls short in certain applications, such as CCA2 constructions, for which a stronger correctness condition, as that given in Definition 1, is needed.*

We will now define a single-message-based notion of indistinguishability for deterministic encryption of block sources, which as proved in [BFO08], is equivalent to both the simulation-based and indistinguishability-based multiple-message notions.

Definition 2 (Deterministic-encryption security [BFO08]). *Let $\mathsf{TDF} = (\mathsf{TDF.KG}, \mathsf{TDF.F}, \mathsf{TDF.F}^{-1})$ be as in Definition 1. We say that TDF is (k,n)-CPA-indistinguishable if for any two (k,n)-sources $\mathcal{S}_1$ and $\mathcal{S}_2$ we have $(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathcal{S}_1)) \stackrel{c}{\equiv} (\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathcal{S}_2))$, where $(\mathsf{ik}, *) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$.*

We say that TDF is (k,n)-CCA2-indistinguishable if for any two (k,n)-sources $\mathcal{S}_0$ and $\mathcal{S}_1$, and any PPT adversary $\mathcal{A}$ the following probability is negligible:

$$\Pr\left[b = b' : \begin{array}{l} (\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda), b \xleftarrow{\$} \{0,1\}, \mathsf{x}^\star \xleftarrow{\$} \mathcal{S}_b, \mathsf{u}^\star := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}^\star) \\ b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Dec}}}(\mathsf{ik}, \mathsf{u}^\star) \end{array}\right] - \frac{1}{2}$$

where on input u*, the decryption oracle* $\mathcal{O}_{\mathsf{Dec}}$ *returns* $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u})$ *if* $\mathsf{u} \neq \mathsf{u}^\star$*, and* $\perp$ *otherwise.*

We remark that considering only CCA2 queries (as opposed to both CCA1 and CCA2 queries) in the CCA2-indistinguishability definition for deterministic encryption is without loss of generality, since the plaintexts are not chosen by the adversary. See [BFO08] for further explanation.

Definition 3 (Computational Diffie-Hellman (CDH) assumption). *Let* G *be a group-generator scheme, which on input* 1^λ *outputs* $(\mathbb{G}, p, g)$*, where* $\mathbb{G}$ *is the description of a group,* p *is the order of the group which is always a prime number and* g *is a generator of the group. We say that* G *is CDH-hard if for any PPT adversary* $\mathcal{A}$*:* $\Pr[\mathcal{A}(\mathbb{G}, p, g, g^{a_1}, g^{a_2}) = g^{a_1 a_2}] = \mathsf{negl}(\lambda)$*, where* $(\mathbb{G}, p, g) \xleftarrow{\$} \mathsf{G}(1^\lambda)$ *and* $a_1, a_2 \xleftarrow{\$} \mathbb{Z}_p$*.*

2.2 Standard Lemmas

Lemma 1 (Chernoff inequality). *Let* $\mathcal{X}_1, \ldots, \mathcal{X}_m$ *be independent Boolean variables each of expected value at least* p*. Then, for all* $\varepsilon > 0$*:*

$$\Pr\left[\frac{1}{m}\sum_{i=1}^{m}\mathcal{X}_i < p - \varepsilon\right] < e^{-2\varepsilon^2 m}.$$

Lemma 2 (Leftover hash lemma [ILL89]). *Let* $\mathcal{X}$ *be a random variable over* X *and* $h : \mathsf{S} \times \mathsf{X} \to \mathsf{Y}$ *be a 2-universal hash function, where* $|\mathsf{Y}| \leq 2^m$ *for some* $m > 0$*. If* $m \leq \mathsf{H}_\infty(\mathcal{X}) - 2\log\left(\frac{1}{\varepsilon}\right)$*, then* $(h(\mathcal{S}, \mathcal{X}), \mathcal{S}) \approx_\varepsilon (\mathcal{U}, \mathcal{S})$*, where* $\mathcal{S}$ *is uniform over* S *and* $\mathcal{U}$ *is uniform over* Y*.*

3 Smooth Recyclable OWFE

We recall the definition of recyclable one-way function with encryption from [GH18]. We adapt the definition to a setting in which the underlying input distribution is not necessarily uniform. We will also define a smoothness notion, which generalizes the one-wayness notion.

Definition 4 (Recyclable one-way function with encryption (OWFE)). *A recyclable* (k, n)*-OWFE scheme consists of the PPT algorithms* K*,* f*,* E_1*,* E_2 *and* D *with the following syntax.*

- $\mathsf{K}(1^\lambda)$*: Takes the security parameter* 1^λ *and outputs a public parameter* pp *(by tossing coins) for a function* $\mathsf{f}(\mathsf{pp}, \cdot)$ *from* n *bits to* ν *bits.*
- $\mathsf{f}(\mathsf{pp}, \mathsf{x})$*: Takes a public parameter* pp *and a preimage* $\mathsf{x} \in \{0,1\}^n$*, and deterministically outputs* $\mathsf{y} \in \{0,1\}^\nu$*.*
- $\mathsf{E}_1(\mathsf{pp}, (i, b); \rho)$*: Takes a public parameter* pp*, an index* $i \in [n]$*, a bit* $b \in \{0,1\}$ *and randomness* ρ*, and outputs a ciphertext* ct*.*[10]

[10] ct is assumed to contain (i, b).

- $\mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, b); \rho)$*: Takes a public parameter* pp*, a value* y*, an index* $i \in [n]$*, a bit* $b \in \{0, 1\}$ *and randomness* ρ*, and outputs a bit* e*. Notice that unlike* E_1*, which does not take* y *as input, the algorithm* E_2 *does take* y *as input.*
- $\mathsf{D}(\mathsf{pp}, \mathsf{ct}, \mathsf{x})$*: Takes a public parameter* pp*, a ciphertext* ct *and a preimage* $\mathsf{x} \in \{0, 1\}^n$*, and deterministically outputs a bit* e*.*

We require the following properties.

- ***Correctness.*** *For any choice of* $\mathsf{pp} \in \mathsf{K}(1^\lambda)$*, any index* $i \in [n]$*, any preimage* $\mathsf{x} \in \{0, 1\}^n$ *and any randomness value* ρ*, the following holds: letting* $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$*,* $b := \mathsf{x}_i$ *and* $\mathsf{ct} := \mathsf{E}_1(\mathsf{pp}, (i, \mathsf{x}_i); \rho)$*, we have* $\mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, \mathsf{x}_i); \rho) = \mathsf{D}(\mathsf{pp}, \mathsf{ct}, \mathsf{x})$*.*
- (k, n)***-One-wayness:*** *For any* (k, n) *source* $\mathcal{S}$ *and any PPT adversary* $\mathcal{A}$*:*

$$\Pr[\mathsf{f}(\mathsf{pp}, \mathcal{A}(\mathsf{pp}, \mathsf{y})) = \mathsf{y}] = \mathsf{negl}(\lambda),$$

  where $\mathsf{pp} \xleftarrow{\$} \mathsf{K}(1^\lambda)$*,* $\mathsf{x} \xleftarrow{\$} \mathcal{S}$ *and* $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$*.*
- ***Security for encryption:*** *For any* $i \in [n]$ *and* $\mathsf{x} \in \{0, 1\}^n$*:*

$$(\mathsf{x}, \mathsf{pp}, \mathsf{ct}, \mathsf{e}) \stackrel{c}{\equiv} (\mathsf{x}, \mathsf{pp}, \mathsf{ct}, \mathsf{e}')$$

  where $\mathsf{pp} \xleftarrow{\$} \mathsf{K}(1^\lambda)$*,* $\rho \xleftarrow{\$} \{0, 1\}^*$*,* $\mathsf{ct} := \mathsf{E}_1(\mathsf{pp}, (i, 1-\mathsf{x}_i); \rho)$*,* $\mathsf{e} := \mathsf{E}_2\big(\mathsf{pp}, \mathsf{f}(\mathsf{pp}, \mathsf{x}), (i, 1-\mathsf{x}_i); \rho\big)$ *and* $\mathsf{e}' \xleftarrow{\$} \{0, 1\}$*.*

Whenever we say an OWFE scheme (without specifying the parameters), we mean $k = n$*.*

Notation 2. *We define* $\mathsf{E}(\mathsf{pp}, \mathsf{y}, (i, b); \rho) \triangleq (\mathsf{E}_1(\mathsf{pp}, (i, b); \rho), \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, b); \rho))$*.*

We will now define the notion of smoothness which extends the one-wayness property to an indistinguishability-based property.

Definition 5 (Smoothness). *Let* $(\mathsf{K}, \mathsf{f}, \mathsf{E}, \mathsf{D})$ *be as in Definition 4. We say that* $(\mathsf{K}, \mathsf{f}, \mathsf{E}, \mathsf{D})$ *is* (k, n)*-smooth if for any two* (k, n)*-sources* $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *we have* $(\mathsf{pp}, \mathsf{f}(\mathsf{pp}, \mathsf{x}_1)) \stackrel{c}{\equiv} (\mathsf{pp}, \mathsf{f}(\mathsf{pp}, \mathsf{x}_2))$*, where* $\mathsf{pp} \xleftarrow{\$} \mathsf{K}(1^\lambda)$*,* $\mathsf{x}_1 \xleftarrow{\$} \mathcal{S}_1$ *and* $\mathsf{x}_2 \xleftarrow{\$} \mathcal{S}_2$*.*

In the full version of this paper, we show that the recyclable OWFE from [GH18] based on CDH is (k, n)-smooth, for any $k \geq \log p + \omega(\log \lambda)$, where p is the order of the underlying CDH-hard group.

4 Strong TDFs from Smooth Recyclable OWFE

In this section we show that recyclable OWFE implies the existence of TDFs with almost-perfect correctness in the sense of Definition 1. This improves the correctness property of [GH18]; see Remark 1. Moreover, we show that if the base recyclable OWFE scheme is smooth (Definition 5), then the resulting TDF satisfies the notions of security for deterministic encryption (Definition 2). We

will then use this statement along with the CDH-based OWFE from [GH18] to obtain the first deterministic-encryption scheme based on CDH. In particular, the existence of deterministic encryption (even) with CPA security from CDH has been open until now.

A central new tool developed in this work is a *mirroring* technique, which we will describe below. As notation, for a matrix $\mathbf{M} \in \mathbb{Z}_2^{k \times n}$, we define $\mathsf{RSum}(\mathbf{M}) \triangleq \mathbf{M}_1 + \cdots + \mathbf{M}_k \in \mathbb{Z}_2^n$, where $\mathbf{M}_i$ for $i \in [k]$ denotes the ith row of $\mathbf{M}$.

Definition 6. *(The mirror Function* Mir*) Let* $(\mathsf{K}, \mathsf{f}, \mathsf{E}_1, \mathsf{E}_2, \mathsf{D})$ *be a recyclable OWFE scheme. For a public parameter* pp*, a value* $\mathsf{x} \in \{0,1\}^n$*, a matrix* $\mathbf{CT} := \begin{pmatrix} \mathsf{ct}_{1,0}, \mathsf{ct}_{2,0}, \ldots, \mathsf{ct}_{n,0} \\ \mathsf{ct}_{1,1}, \mathsf{ct}_{2,1}, \ldots, \mathsf{ct}_{n,1} \end{pmatrix}$ *of ciphertexts outputted by* E_1*, and a vector* $\mathbf{a} \in \{0,1\}^n$*, the function* $\mathsf{Mir}(\mathsf{pp}, \mathsf{x}, \mathbf{CT}, \mathbf{a})$ *outputs a matrix* $\mathbf{M} \in \mathbb{Z}_2^{2 \times n}$*, where* $\mathbf{M} := \begin{pmatrix} \mathsf{b}_{1,0}, \mathsf{b}_{2,0}, \ldots, \mathsf{b}_{n,0} \\ \mathsf{b}_{1,1}, \mathsf{b}_{2,1}, \ldots, \mathsf{b}_{n,1} \end{pmatrix}$ *is formed deterministically and uniquely according to the following two rules:*

1. *for all* $i \in [n]$*:* $\mathsf{b}_{i,\mathsf{x}_i} = \mathsf{D}(\mathsf{pp}, \mathsf{ct}_{i,\mathsf{x}_i}, \mathsf{x})$*; and*
2. $\mathsf{RSum}(\mathbf{M}) = \mathbf{a}$.

Note that the above computation is deterministic and can be done efficiently.

Construction 3 (TDF construction)**.** *We now present our TDF construction.*

Base primitive. *A recyclable OWFE scheme* $\mathcal{E} = (\mathsf{K}, \mathsf{f}, \mathsf{E}, \mathsf{D})$*. Let* Rand *be the randomness space of the algorithm* E*.*

Construction. *The construction is parameterized over two parameters* $n = n(\lambda)$ *and* $r = r(\lambda)$*, where* n *is the input length to the function* $\mathsf{f}(\mathsf{pp}, \cdot)$*, and* r *will be instantiated in the correctness proof. The input space of the TDF is* $\{0,1\}^n$*.*

- $\mathsf{TDF.KG}(1^\lambda)$*:*
  1. *Sample* $\mathsf{pp} \leftarrow \mathsf{K}(1^\lambda)$.
  2. *For each* $h \in [r]$*:*

$$\mathbf{P}_h := \begin{pmatrix} \rho_{1,0}^{(h)}, \rho_{2,0}^{(h)}, \ldots, \rho_{n,0}^{(h)} \\ \rho_{1,1}^{(h)}, \rho_{2,1}^{(h)}, \ldots, \rho_{n,1}^{(h)} \end{pmatrix} \xleftarrow{\$} \mathsf{Rand}^{2 \times n}, \tag{2}$$

$$\mathbf{CT}_h := \begin{pmatrix} \mathsf{E}_1(\mathsf{pp}, (1,0); \rho_{1,0}^{(h)}), \mathsf{E}_1(\mathsf{pp}, (2,0); \rho_{2,0}^{(h)}), \ldots, \mathsf{E}_1(\mathsf{pp}, (n,0); \rho_{n,0}^{(h)}) \\ \mathsf{E}_1(\mathsf{pp}, (1,1); \rho_{1,1}^{(h)}), \mathsf{E}_1(\mathsf{pp}, (2,1); \rho_{2,1}^{(h)}), \ldots, \mathsf{E}_1(\mathsf{pp}, (n,1); \rho_{n,1}^{(h)}) \end{pmatrix}. \tag{3}$$

  3. *For* $h \in [r]$ *sample* $\mathbf{a}_h \xleftarrow{\$} \{0,1\}^n$.
  4. *Form the index key* ik *and the trapdoor key* tk *as follows:*

$$\mathsf{ik} := (\mathsf{pp}, \mathbf{CT}_1, \ldots, \mathbf{CT}_r, \mathbf{a}_1, \ldots, \mathbf{a}_r), \tag{4}$$

$$\mathsf{tk} := (\mathsf{pp}, \mathbf{P}_1, \ldots, \mathbf{P}_r). \tag{5}$$

- $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$*: Parse* ik *as in Equation 4. Set* $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$*. Return*

$$\mathsf{u} := (\mathsf{y}, \mathsf{Mir}(\mathsf{pp}, \mathsf{x}, \mathbf{CT}_1, \mathbf{a}_1), \ldots, \mathsf{Mir}(\mathsf{pp}, \mathsf{x}, \mathbf{CT}_r, \mathbf{a}_r)). \tag{6}$$

– $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u})$:
  1. *Parse* $\mathsf{tk} := (\mathsf{pp}, \mathbf{P}_1, \ldots, \mathbf{P}_r)$ *and parse* $\mathbf{P}_h$ *for* $h \in [r]$ *as in Equation 2.*
  2. *Parse* $\mathsf{u} := (\mathsf{y}, \mathbf{M}_1, \ldots, \mathbf{M}_r)$, *where for all* $h \in [r]$, $\mathbf{M}_h \in \mathbb{Z}_2^{2\times n}$.
  3. *Reconstruct* $\mathsf{x} := \mathsf{x}_1 \cdots \mathsf{x}_n \in \{0,1\}^n$ *bit-by-bit as follows. To recover the ith bit of* x:
     (a) *If for some* $h \in [r]$, $\mathbf{M}_h[i] = \begin{pmatrix} \mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,0);\rho_{i,0}^{(h)}) \\ 1-\mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,1);\rho_{i,1}^{(h)}) \end{pmatrix}$, *set* $\mathsf{x}_i = 0$. *Here* $\mathbf{M}_h[i]$ *denotes the ith column of* $\mathbf{M}_h$.
     (b) *Else, if for some* $h \in [r]$, $\mathbf{M}_h[i] = \begin{pmatrix} 1-\mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,0);\rho_{i,0}^{(h)}) \\ \mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,1);\rho_{i,1}^{(h)}) \end{pmatrix}$, *set* $\mathsf{x}_i = 1$.
     (c) *Otherwise, halt and return* $\perp$.
  4. *Return* x.

Lemma 3 (TDF correctness). *We have*

$$\Pr_{(\mathsf{ik},\mathsf{tk})}[\exists \mathsf{x} \in \{0,1\}^n \text{ s.t. } \mathsf{TDF.F}^{-1}(\mathsf{tk}, (\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}))) \neq \mathsf{x}] \leq \frac{n2^n}{2^r}, \tag{7}$$

where the probability is taken over $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$. *For instance, setting:* $r = n + \omega(\log \lambda)$ *gives a negligible inversion error.*

Lemma 4 (TDF one-wayness and CPA-indistinguishability security). *Assuming* $\mathcal{E}$ *is an OWFE scheme (i.e., an* (n,n)*-OWFE scheme), the TDF* $(\mathsf{TDF.KG}, \mathsf{TDF.F}, \mathsf{TDF.F}^{-1})$ *given in Construction 3 is one-way. That is, for any PPT adversary* $\mathcal{A}$

$$\Pr[\mathcal{A}(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})) = \mathsf{x}] = \mathsf{negl}(\lambda), \tag{8}$$

where $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$ *and* $\mathsf{x} \xleftarrow{\$} \{0,1\}^n$. *Moreover, if* $\mathcal{E}$ *is* (k,n)*-smooth (Definition 5), the constructed TDF is* (k,n)*-CPA-indistinguishable (Definition 2).*

We may now combine the CDH-based OWFE from [GH18] , 3 and 4 to get the first CPA-secure deterministic encryption scheme from CDH.

Corollary 1 (CDH implies deterministic encryption). *Let* G *be a CDH-hard group scheme. For any* $k \geq \log p + \omega(\log \lambda)$ *and any* $n \geq k$ *(where* p *is the order of the underlying group), there exists a* (k,n)*-CPA-indistinguishable deterministic encryption scheme with plaintext size* n *(in bits) and ciphertext size* $\Theta(n^2)$.

4.1 Proof of Correctness: Lemma 3

Proof. We will use notation given in Construction 3. Note that for a given $\mathsf{x} \in \{0,1\}^n$, the inversion succeeds unless there exists an index $i \in [n]$ for which the following bad event happens.

– $\mathsf{Bad}_{\mathsf{x},i}$: for all $h \in [r]$, $\mathbf{a}_h[i] = \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i,0); \rho_{i,0}^{(h)}) + \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i,1); \rho_{i,0}^{(h)}) \in \mathbb{Z}_2$, where $\mathbf{a}_h[i]$ denotes the i'th coordinate of $\mathbf{a}_h \in \{0,1\}^n$.

Since the bits $\mathbf{a}_h[i]$ for all $h \in [r]$ are chosen uniformly at random (independently of pp and $\rho_{i,b}$'s), we have: $\Pr[\mathsf{Bad}_{\mathsf{x},i}] = 2^{-r}$. Doing a union bound over all column $i \in [n]$ gives the probability $n \cdot 2^{-r}$ of an inversion error for a given x. We conclude using a union bound over all $\mathsf{x} \in \{0,1\}^n$. □

4.2 Proof of One-Wayness and CPA Security: Lemma 4

To prove Lemma 4 we first give a simulated way of sampling an index key together with an image point for a target input value.

Definition 7 (Simulated distribution Sim). *Let $\mathcal{E} = (\mathsf{K}, \mathsf{f}, \mathsf{E}, \mathsf{D})$ be the underlying recyclable OWFE scheme. Fix $\mathsf{x} \in \{0,1\}^n$ and let $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$. We define a simulator $\mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})$, which samples a simulated index key $\mathsf{ik}_{\mathrm{sim}}$ with a corresponding simulated TDF image $\mathsf{u}_{\mathrm{sim}}$ for x, as follows. For $h \in [r]$ sample $r^h_{i,b} \xleftarrow{\$} \{0,1\}^*$ for all $(i,b) \in [n] \times \{0,1\}$, and set*

$$(\mathbf{CT}_h, \mathbf{M}_h) \xleftarrow{\$} \begin{pmatrix} \mathsf{E}_1(\mathsf{pp}, (1,0); r^h_{1,0}), \ldots, \mathsf{E}_1(\mathsf{pp}, (n,0); r^h_{n,0}) \\ \mathsf{E}_1(\mathsf{pp}, (1,1); r^h_{1,1}), \ldots, \mathsf{E}_1(\mathsf{pp}, (n,1); r^h_{n,1}) \end{pmatrix} \\ , \begin{pmatrix} \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (1,0); r^h_{1,0}), \ldots, \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (n,0); r^h_{n,0}) \\ \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (1,1); r^h_{1,1}), \ldots, \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (n,1); r^h_{n,1}) \end{pmatrix}. \quad (9)$$

Let

$$\mathsf{ik}_{\mathrm{sim}} := (\mathsf{pp}, \mathbf{CT}_1, \ldots, \mathbf{CT}_r, \mathsf{RSum}(\mathbf{M}_1), \ldots, \mathsf{RSum}(\mathbf{M}_r))$$
$$\mathsf{u}_{\mathrm{sim}} := (\mathsf{y}, \mathbf{M}_1, \ldots, \mathbf{M}_r).$$

Equipped with the above definition, we now give of the proof of Lemma 4.

Proof of Lemma 4. For any distribution $\mathcal{S}$ over $\{0,1\}^n$, we show that the sole security-for-encryption requirement of the recyclable OWFE implies that

$$(\mathsf{x}, \mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})) \stackrel{c}{\equiv} (\mathsf{x}, \mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})), \quad (10)$$

where $\mathsf{x} \xleftarrow{\$} \mathcal{S}$, $\mathsf{pp} \xleftarrow{\$} \mathsf{K}(1^\lambda)$, $(\mathsf{ik}, *) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$ and $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$.

We first show how to use Equation 10 to derive the one-wayness and indistinguishability security of the resulting TDF from the corresponding one-wayness and smoothness of the underlying OWFE scheme, and will then prove Equation 10.

For one-wayness, if there exists an inverter $\mathcal{A}$ that with non-negligible probability can compute x from $(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$ — where $(\mathsf{ik}, *) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$ and $\mathsf{x} \xleftarrow{\$} \{0,1\}^n$ — then Equation 10 implies that with non-negligible probability the adversary $\mathcal{A}$ can compute x from $\mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})$, where $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$. However, this latter violates the one-wayness of f, because the computation of $\mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})$ may be done efficiently with knowledge of pp, n and y.

For indistinguishability security (Definition 2) let $\mathcal{S}_0$ and $\mathcal{S}_1$ be two (k, n) sources and assume that the recyclable OWFE scheme is k-smooth (Definition 5).

Letting $(\mathsf{ik}, *) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$, $\mathsf{x}_0 \xleftarrow{\$} \mathcal{S}_0$, $\mathsf{x}_1 \xleftarrow{\$} \mathcal{S}_1$, $\mathsf{y}_0 := \mathsf{f}(\mathsf{pp}, \mathsf{x}_0)$ and $\mathsf{y}_1 := \mathsf{f}(\mathsf{pp}, \mathsf{x}_1)$, by Equation 10 we have

$$(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}_0)) \overset{c}{\equiv} \mathsf{Sim}(\mathsf{pp}, n, \mathsf{y}_0) \overset{c}{\equiv} \mathsf{Sim}(\mathsf{pp}, n, \mathsf{y}_1) \overset{c}{\equiv} (\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}_1)),$$

where the second indistinguishability follows from the k-smoothness of the recyclable OWFE scheme, which states $(\mathsf{pp}, \mathsf{y}_0) \overset{c}{\equiv} (\mathsf{pp}, \mathsf{y}_1)$.

We are left to prove Equation 10. Fix the distribution $\mathcal{S}$ for which we want to prove Equation 10. To this end, we change the simulator Sim given in Definition 7 to define a new simulator Sim' which on input $\mathsf{Sim}'(\mathsf{pp}, \mathsf{x})$ samples a pair $(\mathsf{ik}'_{\mathrm{sim}}, \mathsf{u}'_{\mathrm{sim}})$ as follows. Let $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$. For all $h \in [r]$, let $\mathbf{CT}_h$ be sampled as in $\mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})$, but with the following modification to $\mathbf{M}_h$:

- Letting $\mathbf{M}_h := \begin{pmatrix} \mathsf{e}^{(h)}_{1,0}, \ldots, \mathsf{e}^{(h)}_{n,0} \\ \mathsf{e}^{(h)}_{1,1}, \ldots, \mathsf{e}^{(h)}_{n,1} \end{pmatrix}$ be formed as in $\mathsf{Sim}(\mathsf{pp}, \mathsf{y})$, for any $i \in [n]$ change $\mathsf{e}^{(h)}_{i,1-\mathsf{x}_i}$ to a random bit (fresh for each index).

Having defined how $\mathbf{CT}_h$ and $\mathbf{M}_h$ are sampled for $h \in [r]$ during $\mathsf{Sim}'(\mathsf{pp}, \mathsf{x})$, form $(\mathsf{ik}'_{\mathrm{sim}}, \mathsf{u}'_{\mathrm{sim}})$ exactly as how $(\mathsf{ik}_{\mathrm{sim}}, \mathsf{u}_{\mathrm{sim}})$ is formed during $\mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})$.

The security-for-encryption requirement of the OWFE scheme implies that $(\mathsf{x}, \mathsf{ik}_{\mathrm{sim}}, \mathsf{u}_{\mathrm{sim}}) \overset{c}{\equiv} (\mathsf{x}, \mathsf{ik}'_{\mathrm{sim}}, \mathsf{u}'_{\mathrm{sim}})$, where $\mathsf{x} \xleftarrow{\$} \mathcal{S}$, $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{x})$, $(\mathsf{ik}_{\mathrm{sim}}, \mathsf{u}_{\mathrm{sim}}) \xleftarrow{\$} \mathsf{Sim}(\mathsf{pp}, n, \mathsf{y})$ and $(\mathsf{ik}'_{\mathrm{sim}}, \mathsf{u}'_{\mathrm{sim}}) \xleftarrow{\$} \mathsf{Sim}'(\mathsf{pp}, \mathsf{x})$. Moreover, it is easy to verify that $(\mathsf{x}, \mathsf{ik}'_{\mathrm{sim}}, \mathsf{u}'_{\mathrm{sim}})$ is identically distributed to $(\mathsf{x}, \mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}))$, where $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$. The proof is now complete.

□

The TDF given in Construction 3 is CPA secure (in a deterministic-encryption sense), but it is not hard to show that the construction is not CCA2 secure. However, we show in the full version of this paper that using techniques of [RS09, KMO10] one may use the TDF of Construction 3 to build another TDF which is CCA2 secure. This upgrading further increases the ciphertext size, resulting in ciphertext size $\Theta(n^3)$ (for the CDH-based instantiation), where n is the plaintext size.

5 Efficient Strong TDFs from Smooth OWFE

The TDF and deterministic encryption presented in Section 4 have the drawback that the output size grows at least quadratically with the input size. The reason behind this blowup is that we had to do "repetitions," resulting in $\Theta(n/2)$ output bits for every single bit of the input. In this section we show how to do away with excessive use of repetition, and to obtain TDFs (and deterministic encryption) whose image/ciphertext size grows linearly with input size. Our main idea involves the use of error-correcting codes, taking advantage of the local inversion property of our basic TDF. As a result, we will obtain the first CPA-secure deterministic encryption scheme with linear ciphertext size based on CDH. We

stress that, even relying on DDH, previous DDH-based deterministic-encryption and TDF schemes resulted in quadratically large ciphertexts.

Definition 8 **($(m,n,d)_2$-Codes).** *We recall the notion of $(m,n,d)_2$ error-correcting codes. Such a code is given by efficiently computable functions* $(\mathsf{Encode}, \mathsf{Decode})$*, where* $\mathsf{Encode} : \{0,1\}^n \to \{0,1\}^m$*, and where*

1. ***Distance.*** *For any two distinct* $\mathsf{x}_1, \mathsf{x}_2 \in \{0,1\}^n$ *we have* $\mathsf{H}_{\mathrm{dst}}(\mathsf{Encode}(\mathsf{x}_1), \mathsf{Encode}(\mathsf{x}_2)) \geq d$*, where* $\mathsf{H}_{\mathrm{dst}}$ *denotes the Hamming distance.*
2. ***Erasure correction.*** *For any* $\mathsf{x} \in \{0,1\}^n$*, letting* $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$*, given any string* $\mathsf{z}' \in \{0,1,\perp\}^m$*, which has at most* $d-1$ $\perp$ *symbols, and whose all non-*$\perp$ *symbols agree with* z*, we have* $\mathsf{Decode}(\mathsf{z}') = \mathsf{x}$*.*
3. ***Error correction.*** *For any* $\mathsf{x} \in \{0,1\}^n$*, letting* $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$*, given any* $\mathsf{z}' \in \{0,1\}^m$ *such that* $\mathsf{H}_{\mathrm{dst}}(\mathsf{z}, \mathsf{z}') < d/2$*, we have* $\mathsf{Decode}(\mathsf{z}') = \mathsf{x}$*.*

We are interested in binary codes with constant rate, constant relative distance, that is: $m = cn$, and $d = c_1 n$. Such codes can be obtained by concatenating codes with constant rate and constant relative distance over large fields — such as Reed-Solomon codes — with codes with constant rate and binary alphabet. See for instance binary Justesen codes [Jus72].

Definition 9 **(rECC code).** *We define a code that suites our purposes, which is the concatenation of an ECC code with a repetition code. Specifically, for a repetition constant t, a t-rECC code* $(\mathsf{Encode}, \mathsf{Decode})$ *consists of* $\mathsf{Encode}\colon \{0,1\}^n \to \{0,1\}^m$*, which is obtained by first applying a $(cn, n, c_1 n)_2$ code and then repeating each bit of the cn bit codeword t times. Thus, $m = tcn$. Note that this code is now a $(tcn, n, tc_1 n)_2$-code.*

Looking ahead, we remark that the use of these repetition codes makes decoding later easier. Specifically, with this repetition, an m bit codeword can be viewed as having *cn blocks* of t bits each. Furthermore, for decoding it is enough to recover one bit per block for at least $cn - c_1 n + 1$ blocks.

In our constructions, it is instructive to think of $c = 200$, $c_1 = 20$ and $t = 9$ for convenience in proofs.[11]

Block index versus bit index. Having codes given as above based on repetition, for a codeword $\mathsf{z} \in \{0,1\}^m$ we talk about a jth block of z for $j \in [m/t]$ to refer to the collections of the bits with indices $\{(j-1)t+1, \ldots, jt\}$.

Construction 4 (TDF construction)**.** *We now describe our TDF construction.*

Base primitive. *A t-rECC code* $(\mathsf{Encode}, \mathsf{Decode})$*, where* $\mathsf{Encode}\colon \{0,1\}^n \to \{0,1\}^m$*, and a recyclable OWFE scheme* $\mathcal{E} = (\mathsf{K}, \mathsf{f}, \mathsf{E}, \mathsf{D})$*, where* f*'s input space is* $\{0,1\}^m$*. We will instantiate the value of constant t in the correctness proof. Let* Rand *be the randomness space of the encapsulation algorithm* E*.*

Construction.

[11] The choices of the constants were made as above so to have slackness in proofs — they have not been optimized for efficiency.

– $\mathsf{TDF.KG}(1^\lambda)$*:*
  1. *Sample* $\mathsf{pp} \leftarrow \mathsf{K}(1^\lambda)$ *and*

$$\mathbf{P} := \begin{pmatrix} \rho_{1,0}, \rho_{2,0}, \ldots, \rho_{m,0} \\ \rho_{1,1}, \rho_{2,1}, \ldots, \rho_{m,1} \end{pmatrix} \xleftarrow{\$} \mathsf{Rand}^{2\times m}, \tag{11}$$

$$\mathbf{CT} := \begin{pmatrix} \mathsf{ct}_{1,0}, \mathsf{ct}_{2,0}, \ldots, \mathsf{ct}_{m,0} \\ \mathsf{ct}_{1,1}, \mathsf{ct}_{2,1}, \ldots, \mathsf{ct}_{m,1} \end{pmatrix}, \tag{12}$$

  where for all $i \in [m]$ *and* $b \in \{0,1\}$, $\mathsf{ct}_{i,b} := \mathsf{E}_1(\mathsf{pp}, (i,b); \rho_{i,b})$.
  2. *Sample* $\mathbf{a} \xleftarrow{\$} \{0,1\}^m$.
  3. *Form the index key* ik *and the trapdoor key* tk *as follows:*

$$\mathsf{ik} := (\mathsf{pp}, \mathbf{a}, \mathbf{CT}) \qquad \mathsf{tk} := (\mathsf{pp}, \mathbf{a}, \mathbf{P}). \tag{13}$$

– $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$*: Parse* $\mathsf{ik} := (\mathsf{pp}, \mathbf{a}, \mathbf{CT})$*. Let* $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$ *and* $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{z})$*. Return*

$$\mathsf{u} := (\mathsf{y}, \mathsf{Mir}(\mathsf{pp}, \mathsf{z}, \mathbf{CT}, \mathbf{a})). \tag{14}$$

– $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u})$*:*
  1. *Parse* $\mathsf{tk} := (\mathsf{pp}, \mathbf{a}, \mathbf{P})$ *and parse* $\mathbf{P}$ *as in Equation* (11)*. Parse* $\mathsf{u} := (\mathsf{y}, \mathbf{M})$*, where* $\mathbf{M} \in \mathbb{Z}_2^{2\times m}$*. If* $\mathsf{RSum}(\mathbf{M}) \neq \mathbf{a}$*, then return* $\perp$*.*
  2. *Construct* $\mathsf{z}' := \mathsf{z}'_1 \cdots \mathsf{z}'_m$ *bit-by-bit as follows. To recover the* i*th bit of* z'*:*
     (a) *If* $\mathbf{M}[i] = \begin{pmatrix} \mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,0);\rho_{i,0}) \\ 1-\mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,1);\rho_{i,1}) \end{pmatrix}$*, set* $\mathsf{z}'_i = 0$*. Here* $\mathbf{M}[i]$ *denotes the* i*th column of* $\mathbf{M}$*.*
     (b) *Else if* $\mathbf{M}[i] = \begin{pmatrix} 1-\mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,0);\rho_{i,0}) \\ \mathsf{E}_2(\mathsf{pp},\mathsf{y},(i,1);\rho_{i,1}) \end{pmatrix}$*, set* $\mathsf{z}'_i = 1$*.*
     (c) *Else, set* $\mathsf{z}'_i = \perp$*.*
  3. *Letting* $\mathsf{x} := \mathsf{Decode}(\mathsf{z}')$*, if* $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}) = \mathsf{u}$*, then return* x*. Otherwise, return* $\perp$*.*

We will now give the correctness and security statements about our TDF, and will prove them in the subsequent subsections.

Lemma 5 (Correctness). *Using a* t*-rECC code* $(\mathsf{Encode}, \mathsf{Decode})$ *with parameters* $(tcn, n, tc_1n)_2$ *(Definition 9), we have*

$$\Pr_{(\mathsf{ik},\mathsf{tk})}[\exists \mathsf{x} \in \{0,1\}^n \text{ s.t. } \mathsf{TDF.F}^{-1}(\mathsf{tk}, (\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}))) \neq \mathsf{x}] \leq 2^n \cdot \mathsf{e}^{-\frac{(2^t c_1 - c)^2 n}{2^{2t-1} c}}. \tag{15}$$

In particular, by choosing the repetition constant t *based on* c *and* c_1 *in such a way that* $2^t c_1 > c$ *and that* $\frac{(2^t c_1 - c)^2}{2^{2t-1} c} \geq 0.7$*, we will have a negligible error.*

Lemma 6 (TDF one-wayness and CPA-indistinguishability security). *Assuming* $\mathcal{E}$ *is an* (n,m)*-OWFE scheme, the TDF* $(\mathsf{TDF.KG}, \mathsf{TDF.F}, \mathsf{TDF.F}^{-1})$ *given in Construction 3 is one-way. That is, for any PPT adversary* $\mathcal{A}$

$$\Pr[\mathcal{A}(\mathsf{ik}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})) = \mathsf{x}] = \mathsf{negl}(\lambda), \tag{16}$$

where $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$ *and* $\mathsf{x} \xleftarrow{\$} \{0,1\}^n$*. Moreover, assuming that the underlying OWFE scheme is* (k,m)*-smooth (Definition 5), the constructed TDF is* (k,n)*-indistinguishable (Definition 2).*

Theorem 5 (CCA2-indistinguishability security). *Assuming that the underlying OWFE scheme is (k, m)-smooth and by appropriately choosing the parameters (in particular we will have $t, c, c_1 \in O(1)$), the constructed TDF is (k, n)-CCA2-indistinguishable.*

We may now combine the CDH-based OWFE from [GH18] , 5 with Theorem 5 to get the following corollary.

Corollary 2 (CDH implies efficient deterministic encryption). *Let G be a CDH-hard group scheme. For any $k \geq \log p + \omega(\log \lambda)$ and any $n \geq k$ (where p is the order of the underlying group), there exists a (k, n)-CCA2-indistinguishable deterministic encryption scheme with plaintext size n (in bits) and ciphertext size $\log p + O(n)$.*

We prove Lemma 5 and 6 in the full version of this paper.

5.1 Proof of CCA2 Security: Theorem 5

We give the proof of Theorem 5 via a series of lemmas. We first start with the following notation.

Notation 6. *For an OWFE scheme $(\mathsf{K}, \mathsf{f}, \mathsf{E}_1, \mathsf{E}_2, \mathsf{D})$, letting $\mathbf{P} := \left(\begin{smallmatrix} \rho_{1,0}, \rho_{2,0}, \ldots, \rho_{m,0} \\ \rho_{1,1}, \rho_{2,1}, \ldots, \rho_{m,1} \end{smallmatrix}\right)$ we define*

$$\mathsf{E}(\mathsf{pp}, \mathsf{y}, \mathbf{P}) \triangleq \begin{pmatrix} \mathsf{E}_1(\mathsf{pp}, (1,0); \rho_{1,0}), \ldots, \mathsf{E}_1(\mathsf{pp}, (m,0); \rho_{m,0}) \\ \mathsf{E}_1(\mathsf{pp}, (1,1); \rho_{1,1}), \ldots, \mathsf{E}_1(\mathsf{pp}, (m,1); \rho_{m,1}) \end{pmatrix}, \\ \begin{pmatrix} \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (1,0); \rho_{1,0}), \ldots, \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (m,0); \rho_{m,0}) \\ \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (1,1); \rho_{1,1}), \ldots, \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (m,1); \rho_{m,1}) \end{pmatrix}.$$

Half-trapdoor keys. In the proof of Theorem 5 we will make use of an alternative way of inversion which works with respect to knowledge of half of all the randomness values that were fixed in the trapdoor key. We refer to such trapdoor keys as *half* trapdoor keys (or simulated trapdoor keys). Recall that a real trapdoor key is of the form

$$(\mathsf{pp}, \mathbf{a}, (\rho_{1,0}, \rho_{1,1}), \ldots, (\rho_{m,0}, \rho_{m,1})). \tag{17}$$

A half-trapdoor key is a reduced version of a full trapdoor key in that we forget one randomness value from each pair, while remembering whether we chose to keep the first or the second coordinate of that pair. Formally, given a full trapdoor key as in Equation (17), a half trapdoor key is obtained based on a string $\mathsf{s} \in \{0,1\}^m$ as $\mathsf{tk}_{\mathrm{rd}} := (\mathsf{pp}, \mathbf{a}, \mathsf{s}, (\rho_1, \ldots, \rho_m))$, where $\rho_i = \rho_{i,\mathsf{s}_i}$. (The subscript rd stands for "reduced.")

We will now define how to perform inversion w.r.t. half-trapdoor keys.

Definition 10 (Half-trapdoor inversion $\mathsf{TDF.F}_{\mathrm{rd}}^{-1}$). *For an image $\mathsf{u} := (\mathsf{y}, \mathbf{M})$ of our constructed TDF and a half-trapdoor key $\mathsf{tk}_{\mathrm{rd}} := (\mathsf{pp}, \mathbf{a}, \mathsf{s}, (\rho_1, \ldots, \rho_m))$ we define $\mathsf{TDF.F}_{\mathrm{rd}}^{-1}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u})$ as follows:*

1. *If $\mathsf{RSum}(\mathbf{M}) \neq \mathbf{a} \in \mathbb{Z}_2^m$, then return $\perp$.*

2. *Construct* $\mathsf{z}' \in \{0,1\}^m$ *bit by bit as follows. For all* $i \in [m]$*, we denote by* $\mathbf{M}[i] = \binom{\mathsf{e}_{i,0}}{\mathsf{e}_{i,1}}$ *the i'th column of* $\mathbf{M}$*. If* $\mathsf{e}_{i,\mathsf{s}_i} = 1 - \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, \mathsf{s}_i); \rho_i)$*, then set* $\mathsf{z}'_i = 1 - \mathsf{s}_i$*; otherwise set* $\mathsf{z}'_i = \perp$*.*
3. *For all* $j \in [cn]$*, if* $\exists i^* \in \{(j-1)t+1, \ldots, jt\}$ *such that* $\mathsf{z}'_{i^*} \neq \perp$ *then for all* $i \in \{(j-1)t+1, \ldots, jt\}$ *set* $\mathsf{z}''_i = \mathsf{z}'_{i^*}$*; else set* $\mathsf{z}''_i = \mathsf{s}_i$
4. *Letting* $\mathsf{x} := \mathsf{Decode}(\mathsf{z}'')$*, if* $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}) = \mathsf{u}$*, return* x*. Otherwise, return* $\perp$*.*

As terminology, we say that $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u})$ *is able to open the ith column of* $\mathbf{M}$ *if* $\mathsf{z}'_i \neq \perp$ *(i.e., if* $\mathsf{e}_{i,\mathsf{s}_i} = 1 - \mathsf{E}_2(\mathsf{pp}, \mathsf{y}, (i, \mathsf{s}_i); \rho_i)$*).*

We first fix some notation and will then prove a useful property about half-inversion simulation, which in turn will be used in the CCA2 proof.

Notation 7 (Half trapdoor keys)**.** *For a given* $\mathsf{tk} := (\mathsf{pp}, \mathbf{a}, (\rho_{1,0}, \rho_{1,1}), \ldots, (\rho_{m,0}, \rho_{m,1}))$ *and* $\mathsf{z} \in \{0,1\}^m$ *we define* $\mathsf{tk}/\mathsf{z} \triangleq (\mathsf{pp}, \mathbf{a}, \mathsf{z}, \rho_{1,\mathsf{z}_1}, \ldots, \rho_{m,\mathsf{z}_m})$*.*

We now give the following lemma about the effectiveness of the half-trapdoor inversion procedure.

Lemma 7 (Half-trapdoor inversion). *Fix* $\mathsf{x} \in \{0,1\}^n$ *and let* $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$*. Using code* $(\mathsf{Encode}, \mathsf{Decode})$ *and setting* t *such that* $1 - 2^{-t} \geq \frac{1}{2} + \frac{c_1}{2c} - \frac{2}{c_1}$*, we have*

$$\Pr_{(\mathsf{ik},\mathsf{tk})}[\exists \mathsf{x}' \in \{0,1\}^n \backslash \{\mathsf{x}\} \text{ s.t. } \mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u}') \neq \mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}')] = 2^n \mathsf{e}^{-\frac{(c_1-4)^2 n}{2c_1}},$$

where $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$*,* $\mathsf{u}' := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')$ *and* $\mathsf{tk}_{\mathrm{rd}} := \mathsf{tk}/\mathsf{z}$*. Thus, by appropriately choosing* c_1 *and* c *(and* t *based on these two values) the above probability will be negligible.*

Proof. Fix $\mathsf{x} \in \{0,1\}^n$ and let $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$. For a sampled $(\mathsf{ik}, \mathsf{tk})$ we define the event Bad as

$$\mathrm{Bad} := \exists \mathsf{x}' \in (\{0,1\}^n \setminus \{\mathsf{x}\}) \text{ s.t. } \mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u}') \neq \mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}'),$$

where $\mathsf{u}' := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')$ and $\mathsf{tk}_{\mathrm{rd}} := \mathsf{tk}/\mathsf{z}$.

First, note that if $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')) = \perp$, then $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')) = \perp$, and if $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')) \neq \perp$, then $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')) = \mathsf{x}' = \mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}'))$. This follows from the descriptions of $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}$ and $\mathsf{TDF.F}^{-1}$, and from the correctness property of $\mathsf{TDF.F}^{-1}$.

Thus, defining

$$\mathrm{Bad}' \colon \exists \mathsf{x}' \in (\{0,1\}^n \setminus \{\mathsf{x}\}) \text{ s.t. } \left(\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, (\mathsf{TDF}(\mathsf{ik}, \mathsf{x}'))) = \perp\right),$$

we have $\Pr[\mathrm{Bad}] \leq \Pr[\mathrm{Bad}']$. In what follows, for any fixed $\mathsf{x}' \in \{0,1\}^n$ we will show

$$\Pr[\mathrm{Bad}_{\mathsf{x}'}] \leq \mathsf{e}^{-\frac{(c_1-4)^2 n}{2c_1}},$$

where we define $\mathrm{Bad}_{\mathsf{x}'} := \mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, (\mathsf{TDF}(\mathsf{ik}, \mathsf{x}'))) = \perp$. This will complete the proof.

For the fixed $\mathsf{x}' \in \{0,1\}^n$, let $\mathsf{u}' := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}')$ and $\mathsf{z}' := \mathsf{Encode}(\mathsf{x}')$. Parse $\mathsf{u}' := (\mathsf{y}', \mathbf{M}')$.

In order to argue about the correctness of the output of $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}')$, let z^* denote the string that is constructed bit-by-bit during the execution of $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}')$. We will show that the fractional distance $\frac{\mathsf{H}_{\mathrm{dst}}(\mathsf{z}^*, \mathsf{z}')}{m} \leq \frac{c_1}{2c}$, and thus by the error-correction property of the underlying code (Item 3 of Definition 8) we will have $\mathsf{TDF.F}^{-1}_{\mathrm{rd}}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}') = \mathsf{x}'$, as desired.

Let $\mathsf{S} \subseteq [cn]$ be the set of block indices on which z and z' are different. (Recall the notion of block index from the paragraphs after Definition 8.) Suppose $|\mathsf{S}| = v$ and let $\mathsf{S} := \{u_1, \ldots, u_v\}$. Note that $v \geq c_1 n$. We have

1. For any block index $j \in [cn] \setminus \mathsf{S}$, all those t bits of z' which come from its jth block will be equal to those of z^*. Namely, for all $i \in \{(j-1)t+1, \ldots, jt\}$ we have $\mathsf{z}'_i = \mathsf{z}^*_i$.
2. For any block index $j \in \mathsf{S}$, if the jth block of z' is different from that of z^*, then *all* the columns of the jth block of $\mathbf{M}'$ are hung; Namely, for all $i \in \{(j-1)t+1, \ldots, jt\}$, $\mathbf{M}'[i]$ is hung. (See Definition **??** for the definition of hung.) This fact follows easily by inspection.

With the above intuition in mind, for $j \in [v]$ let W_j be a Boolean random variable where $\mathsf{W}_j = 0$ if the entire u_j'th block of $\mathbf{M}'$ is hung (i.e., all the corresponding t columns are hung), and $\mathsf{W}_j = 1$, otherwise. Note that for all j: $\Pr[\mathsf{W}_j = 1] = 1 - 2^{-t}$; this follows from the random choice of the vector $\mathbf{a}$ which is fixed in ik. Thus, by the bounds fixed in the lemma we have $\Pr[\mathsf{W}_j = 1] \geq \frac{1}{2} + \frac{c_1}{2c} - \frac{2}{c_1}$. Let $p := 1 - 2^{-t}$. We have

$$\Pr[\mathrm{Bad}_{\mathsf{x}'}] \leq \Pr[\frac{1}{v}\sum_{j=1}^{v} \mathsf{W}_j < \frac{c_1}{2c}] \leq \Pr[\frac{1}{v}\sum_{j=1}^{v} \mathsf{W}_j < p - (\frac{1}{2} - \frac{2}{c_1})] \leq^* \mathsf{e}^{-2(\frac{1}{2}-\frac{2}{c_1})^2 v}$$
$$\leq \mathsf{e}^{-2(\frac{1}{2}-\frac{2}{c_1})^2 c_1 n} = \mathsf{e}^{-\frac{2(c_1-4)^2 c_1 n}{4c_1^2}} = \mathsf{e}^{-\frac{(c_1-4)^2 n}{2c_1}}, \quad (18)$$

where the probability marked with $*$ follows from the Chernoff bounds. The proof is now complete.

□

Our CCA2 hybrids will also make use of a simulated way of producing index/trapdoor keys. This procedure is described below.

Definition 11 (Simulated TDF key generation). *We define a simulated key-generation algorithm for the TDF given in Construction 4. Let* $(\mathsf{K}, \mathsf{f}, \mathsf{E}_1, \mathsf{E}_2, \mathsf{D})$ *be the underlying OWFE scheme. The simulated key generation algorithm* $\mathsf{TDF.KG}_{\mathrm{sim}}(\mathsf{pp}, \mathsf{y})$ *takes* pp *and an image* y *of the function* f *as input, and outputs* $(\mathsf{ik}, \mathsf{tk})$ *formed as follows. Sample* $\mathbf{P} \xleftarrow{\$} \mathsf{Rand}^{2\times m}$ *and set* $(\mathbf{CT}, \mathbf{M}) := \mathsf{E}(\mathsf{pp}, \mathsf{y}, \mathbf{P})$. *(See Notation 6.) Set* $\mathsf{ik} := (\mathsf{pp}, \mathsf{RSum}(\mathbf{M}), \mathbf{CT})$ *and* $\mathsf{tk} := (\mathsf{pp}, \mathsf{RSum}(\mathbf{M}), \mathbf{P})$.

We will now describe the hybrids for proving CCA2 security of the deterministic encryption scheme. We define the hybrids with respect to a distribution $\mathcal{D}$ and will then instantiate the distribution in the subsequent lemmas.

Hybrid $\mathsf{H}_0[\mathcal{D}]$*: real game.*

- **Index/trapdoor keys.** Sample $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$.
- **Challenge ciphertext.** Set $\mathsf{u} := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$, where $\mathsf{x} \leftarrow \mathcal{D}$.
- **CCA2 inversion queries.** Reply to each inversion query $\mathsf{u}' \neq \mathsf{u}$ with $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u}')$.

Hybrid $\mathsf{H}_1[\mathcal{D}]$*: half-trapdoor inversion.* Same as H_0 except we reply to inversion queries using a half trapdoor and by using the algorithm $\mathsf{TDF.F}_{\mathrm{rd}}^{-1}$.

- **Index/trapdoor keys.** Sample $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$. Set the index key to be ik and form the trapdoor key as follows: sample $\mathsf{x} \leftarrow \mathcal{D}$, let $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$ and set the trapdoor key to be $\mathsf{tk}_{\mathrm{rd}} := \mathsf{tk}/\mathsf{z}$ (Notation 7).
- **Challenge ciphertext.** Return $\mathsf{u} := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$, where recall that x was sampled in the previous step.
- **CCA2 inversion queries.** Reply to each inversion query $\mathsf{u}' \neq \mathsf{u}$ with $\mathsf{TDF.F}_{\mathrm{rd}}^{-1}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}')$.

Hybrid $\mathsf{H}_2[\mathcal{D}]$*: half-trapdoor inversion with a simulated index key.* Same as $\mathsf{H}_1[\mathcal{D}]$ except that we sample the index key and the challenge ciphertext jointly in a simulated way.

- **Index/trapdoor keys:**
  1. Sample $\mathsf{x} \leftarrow \mathcal{D}$, and let $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$. Set $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{z})$.
  2. Sample $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}_{\mathrm{sim}}(\mathsf{pp}, \mathsf{y})$.
  3. Set the index key to be ik and the trapdoor key to be $\mathsf{tk}_{\mathrm{rd}} := \mathsf{tk}/\mathsf{z}$.
- **Challenge ciphertext.** Return $\mathsf{u} := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$, where recall that x was sampled above.
- **CCA2 inversion queries.** Reply to each inversion query $\mathsf{u}' \neq \mathsf{u}$ with $\mathsf{TDF.F}_{\mathrm{rd}}^{-1}(\mathsf{tk}_{\mathrm{rd}}, \mathsf{u}')$.

Hybrid $\mathsf{H}_3[\mathcal{D}]$*: Full trapdoor inversion with a simulated index key.* Same as $\mathsf{H}_2[\mathcal{D}]$ except we use tk as the trapdoor key (instead of $\mathsf{tk}_{\mathrm{rd}}$) and will reply to each CCA2 inversion query $\mathsf{u}' \neq \mathsf{u}$ with $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u}')$. That is:

- **Index/trapdoor keys:**
  1. Sample $\mathsf{x} \leftarrow \mathcal{D}$, and let $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$. Set $\mathsf{y} := \mathsf{f}(\mathsf{pp}, \mathsf{z})$.
  2. Let the index/trapdoor key be $(\mathsf{ik}, \mathsf{tk}) \xleftarrow{\$} \mathsf{TDF.KG}_{\mathrm{sim}}(\mathsf{pp}, \mathsf{y})$.
- **Challenge ciphertext.** Return $\mathsf{u} := \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})$.
- **CCA2 inversion queries.** Reply to each inversion query $\mathsf{u}' \neq \mathsf{u}$ with $\mathsf{D}(\mathsf{tk}, \mathsf{u}')$.

The above concludes the description of the hybrids. We now define some notation and will then prove some lemmas.

Notation. For $i \in \{0,1,2,3\}$ we use $\mathsf{out}_i[\mathcal{D}]$ to denote the output bit of an underlying adversary in hybrid $\mathsf{H}_i[\mathcal{D}]$. For $i,j \in \{0,1,2,3\}$ and two distributions $\mathcal{S}_0$ and $\mathcal{S}_1$, we write $\mathsf{H}_i[\mathcal{S}_0] \overset{c}{\equiv} \mathsf{H}_j[\mathcal{S}_1]$ to mean that for all PPT adversaries $\mathcal{A}$ we have $|\Pr[\mathsf{out}_i[\mathcal{S}_0]=1] - \Pr[\mathsf{out}_j[\mathcal{S}_1]=1]| = \mathsf{negl}(\lambda)$.

The proof of Theorem 5 follows from the following lemmas.

Lemma 8 (Indistinguishability of Hybrids H_0 and H_1). *By appropriately choosing the parameters for c, c_1 and t, for any PPT adversary $\mathcal{A}$ we have $|\Pr[\mathsf{out}_0[\mathcal{D}]=1] - \Pr[\mathsf{out}_1[\mathcal{D}]=1]| = \mathsf{negl}(\lambda)$.*

Lemma 9 (Indistinguishability of Hybrids H_1 and H_2). *If the OWFE satisfies the security-for-encryption property, then for any distribution $\mathcal{D}$ and any PPT adversary $\mathcal{A}$, we have $|\Pr[\mathsf{out}_1[\mathcal{D}]=1] - \Pr[\mathsf{out}_2[\mathcal{D}]=1]| = \mathsf{negl}(\lambda)$.*

Lemma 10 (Indistinguishability of Hybrids H_2 and H_3). *If the OWFE satisfies the security-for-encryption property and by choosing the parameters appropriately, then for any distribution $\mathcal{D}$ and any PPT adversary $\mathcal{A}$, we have $|\Pr[\mathsf{out}_2[\mathcal{D}]=1] - \Pr[\mathsf{out}_3[\mathcal{D}]=1]| = \mathsf{negl}(\lambda)$.*

Lemma 11 (CCA2 Security in H_3). *If the OWFE is (k,m)-smooth, then for any two (k,n) sources $\mathcal{S}_0$ and $\mathcal{S}_1$ and any PPT adversary $\mathcal{A}$, we have $|\Pr[\mathsf{out}_3[\mathcal{S}_0]=1] - \Pr[\mathsf{out}_3[\mathcal{S}_1]=1]| = \mathsf{negl}(\lambda)$.*

Proof of Theorem 5. By applying the above lemmas, for any (k,n)-sources $\mathcal{S}_0$ and $\mathcal{S}_1$, we have:

$$\mathsf{H}_0[\mathcal{S}_0] \overset{c}{\equiv} \mathsf{H}_1[\mathcal{S}_0] \overset{c}{\equiv} \mathsf{H}_2[\mathcal{S}_0] \overset{c}{\equiv} \mathsf{H}_3[\mathcal{S}_0] \overset{c}{\equiv} \mathsf{H}_3[\mathcal{S}_1] \overset{c}{\equiv} \mathsf{H}_2[\mathcal{S}_1] \overset{c}{\equiv} \mathsf{H}_1[\mathcal{S}_1] \overset{c}{\equiv} \mathsf{H}_0[\mathcal{S}_1].$$

We prove these lemmas in the full version of the paper.

6 Lossy TDFs with Linear-Image Size

In this section, using our erasure-resilient code techniques, we show how to adapt a variant of the TDFs from [PW08, PW11] to obtain the first lossy trapdoor functions with images growing linearly in their inputs, based on the DDH assumption. This improves upon the lossy TDFs from [PW08, PW11], whose output size is quadratic in the input size. We first recall the definition of lossy TDF from [PW08, PW11].

Definition 12 (Lossy TDFs [PW08, PW11]). *An (n,k)-lossy TDF ((n,k)-LTDF) is given by four PPT algorithms $\mathsf{TDF.KG}$, $\mathsf{TDF.KG_{ls}}$, $\mathsf{TDF.F}$, $\mathsf{TDF.F^{-1}}$, where $\mathsf{TDF.KG_{ls}}(1^\lambda)$ only outputs a single key (as opposed to a pair of keys), and where the following properties hold:*

- ***Correctness in real mode.*** *The TDF $(\mathsf{TDF.KG}, \mathsf{TDF.F}, \mathsf{TDF.F^{-1}})$ satisfies correctness in the sense of Definition 1.*

– ***k*-Lossiness.** *For all but negligible probability over the choice of* $\mathsf{ik}_{\mathsf{ls}} \xleftarrow{\$} \mathsf{TDF.KG}_{\mathsf{ls}}(1^\lambda)$*, we have* $|\mathsf{TDF.F}(\mathsf{ik}_{\mathsf{ls}}, \{0,1\}^n)| \leq 2^k$*, where we use* $\mathsf{TDF.F}(\mathsf{ik}_{\mathsf{ls}}, \{0,1\}^n)$ *to denote the set of all images of* $\mathsf{TDF.F}(\mathsf{ik}_{\mathsf{ls}}, \cdot)$*.*
– ***Indistinguishability of real and lossy modes.*** *We have* $\mathsf{ik} \stackrel{c}{\equiv} \mathsf{ik}_{\mathsf{ls}}$*, where* $(\mathsf{ik}, *) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$ *and* $\mathsf{ik}_{\mathsf{ls}} \xleftarrow{\$} \mathsf{TDF.KG}_{\mathsf{ls}}(1^\lambda)$*.*

Lossiness rate. In the definition above, we refer to the fraction $1 - k/n$ as the *lossiness rate*, describing the fraction of the bits lost. Ideally, we want this fraction to be as close to 1 as possible, e.g., $1 - o(1)$.

Our LTDF construction makes use of a balanced predicate, defined below.

6.1 Lossy TDF from DDH

Our LTDF construction makes use of the following notation.

Notation. Letting $\mathsf{x} \in \{0,1\}^n$ and $\mathbf{M} := \left(\begin{smallmatrix} g_{1,0}, g_{2,0} \cdots, g_{n,0} \\ g_{1,1}, g_{2,1}, \ldots, g_{n,1} \end{smallmatrix}\right)$ we define $\mathsf{x} \odot \mathbf{M} = \prod_{j \in [n]} g_{j,\mathsf{x}_j}$. For $i \in [n]$, $b \in \{0,1\}$ and $\mathbf{M}$ as above, we define the matrix $\mathbf{M}' := (\mathbf{M} \xrightarrow[(i,b)]{} g')$ to be the same as $\mathbf{M}$ except that instead of $g_{i,b}$ we put g' in $\mathbf{M}'$. If $\mathbf{M}$ is matrix of group elements, then $\mathbf{M}^r$ denotes entry-wise exponentiation to the power of r.

Overview of the construction and techniques. Let us first demonstrate the idea for retrieving the first bit of the input. Imagine two $2 \times n$ matrices $\mathbf{M}$ and $\mathbf{M}'$, where $\mathbf{M} := \left(\begin{smallmatrix} g_{1,0}, \ldots, g_{n,0} \\ g_{1,1}, \ldots, g_{n,1} \end{smallmatrix}\right)$ is chosen at random and where $\mathbf{M}' := (\mathbf{M}^r \xrightarrow[(1,b)]{} g_1)$, where $r \xleftarrow{\$} \mathbb{Z}_p$, $b \xleftarrow{\$} \{0,1\}$ and $g_1 \xleftarrow{\$} \mathbb{G}$. That is, $\mathbf{M}'$ is a perturbed rth power of $\mathbf{M}$, in that we replace one of the two elements of the first column of the exponentiated matrix with a random group element.

Think of $(\mathbf{M}, \mathbf{M}')$ as the index key. Suppose an evaluator $\mathsf{TDF.F}$ with input $\mathsf{x} \in \{0,1\}^n$ wants to use $(\mathbf{M}, \mathbf{M}')$ to communicate her first bit x_1 to an inverter who has knowledge of b, g_1 and r. A first attempt for $\mathsf{TDF.F}$ would be to output two group elements $(\tilde{g}, g_1') := (\mathsf{x} \odot \mathbf{M}, \mathsf{x} \odot \mathbf{M}')$. Given $(\tilde{g}, g_1')$, if $g_1' = \tilde{g}^r$, then $\mathsf{x}_1 = 1 - b$; otherwise, $\mathsf{x}_1 = b$ — hence allowing $\mathsf{TDF.F}^{-1}$ to recover x_1.[12]

The above method is in fact what used (implicitly) in all previous approaches [PW08, FGK+10, PW11]. However, the cost paid is high: for communicating one bit of information we need to output (at least) one group element.

We will now illustrate our main idea. Let $\mathsf{BL} : \mathbb{G} \to \{0,1\}$ be a *balanced* predicate, meaning that $\mathsf{BL}(g^*)$ on a randomly generated g^* is a completely random bit. (We will show how to build this object unconditionally.) Returning to the above idea, instead of sending $(\tilde{g}, g_1')$ we will send $(\tilde{g}, \mathsf{BL}(g_1')) \in (\mathbb{G}, \{0,1\})$. Before arguing correctness and security, note that this method yields linear image

[12] For simplicity assume $g_1 \neq g_{1,b}^r$, hence we will not have a hung situation.

size for the whole input, because the group element $\tilde{g}$ can be re-used across all indices.

To argue correctness, let us see how $\mathsf{TDF.F}^{-1}$ — given b, r and g_1 — may invert an encoding $(\tilde{g}, b')$ of the first bit of x. To this end, note the following two facts:

1. If $\mathsf{x}_1 = 1 - b$, then $b' = \mathsf{BL}(\tilde{g}^r)$.
2. If $\mathsf{x}_1 = b$, then $b' = \mathsf{BL}(\frac{\tilde{g}^r g_1}{g_{1,b}^r})$.

Thus, if $\mathsf{BL}(\tilde{g}^r) \neq \mathsf{BL}(\frac{\tilde{g}^r g_1'}{g_{1,b}^r})$, then we can determine the value of x_1. This is because in this case we either have

- $b' = \mathsf{BL}(\tilde{g}^r)$ and $b' \neq \mathsf{BL}(\frac{\tilde{g}^r g_1}{g_{1,b}^r})$: which implies $\mathsf{x}_1 = 1 - b$;
- $b' \neq \mathsf{BL}(\tilde{g}^r)$ and $b' = \mathsf{BL}(\frac{\tilde{g}^r g_1}{g_{1,b}^r})$: which implies $\mathsf{x}_1 = b$.

Summing up the above discussion, we are unable to determine the value of x_1 only when $\mathsf{BL}(\tilde{g}^r) = \mathsf{BL}(\frac{\tilde{g}^r g_1}{g_{1,b}^r})$. This happens with probability $1/2$ because the predicate BL is balanced. For any constant c, we may reduce this probability to $(1/2)^c$ via repetition for c times. Thus, by choosing the constant c appropriately, and doing the above procedure for every index, we will be able to retrieve a good fraction of all the bits of x, which will make the rest retrievable using error correction.

For security, we will show that this method admits a simple lossy way of generating public keys.

We now formally define the notion of balanced predicates, which will be used in our LTDF construction.

Definition 13 (Balanced predicates). *We say a randomized predicate* $\mathsf{P} : \mathsf{S} \times \{0,1\}^* \to \{0,1\}$ *is balanced over set* S *if* $\Pr[\mathsf{P}(x; r) = 0] = 1/2$*, where* $x \xleftarrow{\$} \mathsf{S}$ *and* $r \xleftarrow{\$} \{0,1\}^*$*.*

In the above definition, if $\mathsf{S} = \{0,1\}^n$, then we have a trivial predicate, one which returns, say, the first bit of its input. For our LTDF construction, we require the existence of a predicate for the underlying group $\mathbb{G}$. Assuming any 1-1 mapping $\mathbb{G} \to \{0,1\}^n$ (which may not be surjective), we may define the predicate P as the inner product function over F_2: i.e., $\mathsf{P}(x, r) = \langle x, r \rangle$.

Construction 8 (Linear-image lossy TDF)**.** *Let* G *be a group scheme and let* $(\mathsf{Encode}, \mathsf{Decode})$ *be an erasure code, where* $\mathsf{Encode}\colon \{0,1\}^n \to \{0,1\}^m$ *(Definition 8). Also, let* BL *be a balanced predicate for the underlying group (Definition 13).*

We define our LTDF construction $(\mathsf{TDF.KG}, \mathsf{TDF.KG_{ls}}, \mathsf{TDF.F}, \mathsf{TDF.F}^{-1})$ *as follows.*

- $\mathsf{TDF.KG}(1^\lambda)$*:*

1. *Sample* $(\mathbb{G}, p, g) \xleftarrow{\$} \mathsf{G}(1^\lambda)$*, and*

$$\mathbf{M} := \begin{pmatrix} g_{1,0}, g_{2,0}, \ldots, g_{m,0} \\ g_{1,1}, g_{2,1}, \ldots, g_{m,1} \end{pmatrix} \xleftarrow{\$} \mathbb{G}^{2\times m}. \tag{19}$$

2. *For all* $i \in [m]$*, sample* $g_i \xleftarrow{\$} \mathbb{G}$*,* $\rho_i \xleftarrow{\$} \mathbb{Z}_p$ *and* $b_i \xleftarrow{\$} \{0,1\}$*.*
3. *Sample random coins* r *for the underlying function* BL*.*
4. *Set the index and trapdoor keys as*

$$\mathsf{ik} := (\mathbf{M}, (\mathbf{M}^{\rho_1} \xrightarrow[(1,b_1)]{} g_1), \ldots, (\mathbf{M}^{\rho_m} \xrightarrow[(m,b_m)]{} g_m), r) \tag{20}$$

$$\mathsf{tk} := (\mathbf{M}, (\rho_1, b_1, g_1), \ldots, (\rho_m, b_m, g_m), r). \tag{21}$$

– $\mathsf{TDF.KG_{ls}}(1^\lambda)$*: Return* $\mathsf{ik_{ls}} := (\mathbf{M}, \mathbf{M}^{\rho_1}, \ldots, \mathbf{M}^{\rho_m}, r)$*, where* $\mathbf{M}$ *and* ρ_i *for* $i \in [m]$ *and* r *are sampled as above.*
– $\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x} \in \{0,1\}^n)$*: Parse* $\mathsf{ik} := (\mathbf{M}, \mathbf{M}_1, \ldots, \mathbf{M}_m, r)$*. Set* $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$ *and return*

$$\mathsf{u} := (\mathsf{z} \odot \mathbf{M}, \mathsf{BL}(\mathsf{z} \odot \mathbf{M}_1; r), \ldots, \mathsf{BL}(\mathsf{z} \odot \mathbf{M}_m; r)), \tag{22}$$

– $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{u})$*:*
  1. *Parse* $\mathsf{tk} := (\mathbf{M}, (\rho_1, b_1, g_1), \ldots, (\rho_m, b_m, g_m), r)$ *and* $\mathsf{u} := (g_c, b'_1, \ldots, b'_m)$*. Parse* $\mathbf{M}$ *as in Equation 19.*
  2. *Construct* $\mathsf{z}' := \mathsf{z}'_1 \cdots \mathsf{z}'_m \in \{0, 1, \perp\}^m$ *as follows. For* $i \in [m]$*:*
     (a) *Set* $g'_i := g_c^{\rho_i}$ *and* $g''_i := \frac{g_c^{\rho_i}}{g_{i,b_i}^{\rho_i}} \cdot g_i$*. Then*
        i. *If* $\mathsf{BL}(g'_i; r) = \mathsf{BL}(g''_i; r)$*, set* $\mathsf{z}'_i = \perp$*;*
        ii. *Else, if* $b'_i = \mathsf{BL}(g'_i; r)$*, set* $\mathsf{z}_i = 1 - b_i$*. Else (i.e.,* $b'_i = \mathsf{BL}(g''_i; r)$*), set* $\mathsf{z}_i = b_i$*.*
  3. *Return* $\mathsf{Decode}(\mathsf{z}')$*.*

The following theorem gives the lossiness property of the scheme.

Theorem 9 (Linear-image LTDF from DDH). *Using code* $(\mathsf{Encode}, \mathsf{Decode})$ *such that* $4^t c_1 > 3^t c$ *and if* $\frac{(4^t c_1 - 3^t c)^2}{2^{4t-1} c} \geq 0.7$*, the LTDF of Construction 8 is* $(n, \log p)$*-lossy with image size* $\log p + cn \in \Theta(n)$*. By setting* $n \in \omega(\log p)$ *we obtain* $1 - o(1)$ *lossiness rate.*

We prove all the required properties below.

Lemma 12 ($\log p$**-Lossiness).** *For any* $\mathsf{ik_{ls}} \in \mathsf{TDF.KG_{ls}}(1^\lambda)$ *we have*

$$|\mathsf{TDF.F}(\mathsf{ik_{ls}}, \{0,1\}^n)| \leq p,$$

where recall that p *is the order of the underlying group.*

Proof. Parse $\mathsf{ik_{ls}} := (\mathbf{M}, \mathbf{M}_1, \ldots, \mathbf{M}_m)$. It is easy to verify that for any two messages $\mathsf{x}, \mathsf{x}' \in \{0,1\}^n$ we have

$$\mathsf{TDF.F}(\mathsf{ik_{ls}}, \mathsf{x}) \neq \mathsf{TDF.F}(\mathsf{ik_{ls}}, \mathsf{x}') \iff \mathsf{x} \odot \mathbf{M} \neq \mathsf{x}' \odot \mathbf{M}. \tag{23}$$

The statement of the lemma now follows, since $\{\mathsf{x} \odot \mathbf{M} \mid \mathsf{x} \in \{0,1\}^n\} \subseteq \mathbb{G}$, and thus we have $|\{\mathsf{x} \odot \mathbf{M} \mid \mathsf{x} \in \{0,1\}^n\}| \leq p$. □

Lemma 13 (Indistinguishability of real and lossy modes). *We have* $\mathsf{ik} \overset{c}{\equiv} \mathsf{ik}_{\mathsf{ls}}$*, where* $(\mathsf{ik}, *) \xleftarrow{\$} \mathsf{TDF.KG}(1^\lambda)$ *and* $\mathsf{ik}_{\mathsf{ls}} \xleftarrow{\$} \mathsf{TDF.KG}_{\mathsf{ls}}(1^\lambda)$.

Proof. Immediate by the DDH assumption using standard techniques. □

Lemma 14 (Correctness in real mode). *Using code* $(\mathsf{Encode}, \mathsf{Decode})$*, we have*

$$\Pr_{(\mathsf{ik},\mathsf{tk})}[\exists \mathsf{x} \in \{0,1\}^n \text{ s.t. } \mathsf{TDF.F}^{-1}(\mathsf{tk}, (\mathsf{TDF.F}(\mathsf{ik}, \mathsf{x}))) \neq \mathsf{x}] \leq 2^n \cdot \mathsf{e}^{-\frac{(2^t c_1 - c)^2}{2^{2t-1} c} n}. \quad (24)$$

In particular, by choosing the repetition constant t *such that* $2^t c_1 > c$ *and that* $\frac{(2^t c_1 - c)^2}{2^{2t-1} c} \geq 0.7$*, the probability in Equation (24) will be negligible.*

Proof. Fix $\mathsf{x} \in \{0,1\}^n$ and let $\mathsf{z} := \mathsf{Encode}(\mathsf{x})$. All probabilities below are taken over the random choice of $(\mathsf{ik}, \mathsf{tk})$. Parse

$$\mathsf{tk} := \left(\mathbf{M} := \begin{pmatrix} g_{1,0}, g_{2,0}, \ldots, g_{m,0} \\ g_{1,1}, g_{2,1}, \ldots, g_{m,1} \end{pmatrix}, (\rho_1, b_1, g_1), \ldots, (\rho_m, b_m, g_m), r \right).$$

For input $\mathsf{x} \in \{0,1\}^n$, let $\mathsf{Fail}_{\mathsf{x}}$ be the event that $\mathsf{TDF.F}^{-1}(\mathsf{tk}, \mathsf{TDF.F}(\mathsf{ik}, \mathsf{x})) \neq \mathsf{x}$. Fix $\mathsf{x} \in \{0,1\}^n$ and let $\mathsf{z} := \mathsf{Encode}(\mathsf{x}) \in \{0,1\}^m$. Also, let $\mathsf{u} := \mathsf{TDF}(\mathsf{ik}, \mathsf{x}) := (g_c, b'_1, \ldots, b'_m)$.

Recall that z consists of cn blocks, where each block consists of t identical bits (Definition 9). For each block index $j \in [cn]$ we define an event Bad_j, which corresponds to the event that the inversion algorithm fails to recover the bit that corresponds to the jth block. That is, Bad_j occurs if all the t repetitions inside block j leads to failure during inversion. More formally:

- Bad_j: The event that for all $i \in \{(j-1)t+1, \ldots, jt\}$: $\mathsf{BL}(g'_i; r) = \mathsf{BL}(g''_i; r)$, where $g'_i := g_c^{\rho_i}$ and $g''_i := \frac{g_c^{\rho_i}}{g_{i,b_i}^{\rho_i}} \cdot g_i$.

Note that all Bad_j are i.i.d. events and we have $\Pr[\mathsf{Bad}_j] = (1/2)^t$. The reason for this is that all of $(g_1, \ldots, g_n)$ are sampled uniformly at random independently of all other values, and thus the two group elements g'_i and g''_i are uniform and independent.

Let $\mathsf{Good}_j = \overline{\mathsf{Bad}_j}$ and note that $\Pr[\mathsf{Good}_j] = 1 - (1/2)^t$. We now have

$$\begin{aligned} \Pr[\mathsf{Fail}_{\mathsf{x}}] &\leq \Pr[\sum_{j=1}^{cn} \mathsf{Good}_j \leq cn - c_1 n] = \Pr[\frac{1}{cn} \sum_{j=1}^{cn} \mathsf{Good}_j \leq 1 - \frac{c_1}{c}] \\ &= \Pr[\frac{1}{cn} \sum_{j=1}^{cn} \mathsf{Good}_j \leq 1 - \frac{1}{2^t} - (\frac{c_1}{c} - \frac{1}{2^t})] \leq^* \mathsf{e}^{-2cn(\frac{2^t c_1 - c}{2^t c})^2} \leq \mathsf{e}^{-\frac{(2^t c_1 - c)^2}{2^{2t-1} c} n}, \end{aligned} \quad (25)$$

where the inequality marked with * follows from the Chernoff inequality (Theorem 1 with $p = 1 - 1/2^t$ and $\varepsilon = c_1/c - 1/2^t$. Note that since we must have $\epsilon > 0$, we should have $2^t c_1 > c$.)

We conclude using a union bound over all $\mathsf{x} \in \{0,1\}^n$. □

7 Acknowledgements.

We thank Xiao Liang for suggesting a simplification to Construction 3. We are also grateful to the anonymous reviewers for their comments. Research supported in part from DARPA/ARL SAFEWARE Award W911NF15C0210, AFOSR Award FA9550-15-1-0274, AFOSR YIP Award, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley), and a Google PhD fellowship. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies.

References

BBN+09. M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek. Hedged public-key encryption: How to protect against bad randomness. In *ASIACRYPT 2009*, *LNCS* 5912, pages 232–249, Tokyo, Japan, December 6–10, 2009. Springer, Heidelberg, Germany. 2

BBO07. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In *CRYPTO 2007*, *LNCS* 4622, pages 535–552, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany. 2, 4

BCPT13. E. Birrell, K.-M. Chung, R. Pass, and S. Telang. Randomness-dependent message security. In *TCC 2013*, *LNCS* 7785, pages 700–720, Tokyo, Japan, March 3–6, 2013. Springer, Heidelberg, Germany. 2

BFO08. A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *CRYPTO 2008*, *LNCS* 5157, pages 335–359, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany. 2, 3, 4, 6, 10, 11

BFOR08. M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *CRYPTO 2008*, *LNCS* 5157, pages 360–378, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany. 2, 4

BHY09. M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *EUROCRYPT 2009*, *LNCS* 5479, pages 1–35, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany. 2

BLSV18. Z. Brakerski, A. Lombardi, G. Segev, and V. Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In *EUROCRYPT 2018, Part I*, *LNCS* 10820, pages 535–564, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. 9

BS11. Z. Brakerski and G. Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In *CRYPTO 2011*, *LNCS* 6841, pages 543–560, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Heidelberg, Germany. 4

BW10. X. Boyen and B. Waters. Shrinking the keys of discrete-log-type lossy trapdoor functions. In *ACNS 10*, *LNCS* 6123, pages 35–52, Beijing, China, June 22–25, 2010. Springer, Heidelberg, Germany. 9

CDG+17. C. Cho, N. Döttling, S. Garg, D. Gupta, P. Miao, and A. Polychroniadou. Laconic oblivious transfer and its applications. In *CRYPTO 2017, Part II*, *LNCS* 10402, pages 33–65, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 9

DG17a. N. Döttling and S. Garg. From selective IBE to full IBE and selective HIBE. In *TCC 2017, Part I*, *LNCS* 10677, pages 372–408, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany. 9

DG17b. N. Döttling and S. Garg. Identity-based encryption from the Diffie-Hellman assumption. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 537–569, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 9

DGHM18. N. Döttling, S. Garg, M. Hajiabadi, and D. Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In *PKC 2018, Part I*, *LNCS* 10769, pages 3–31, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany. 9

DH76. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. 1

FGK+10. D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. In *PKC 2010*, *LNCS* 6056, pages 279–295, Paris, France, May 26–28, 2010. Springer, Heidelberg, Germany. 3, 8, 9, 24

GH18. S. Garg and M. Hajiabadi. Trapdoor functions from the computational Diffie-Hellman assumption. LNCS, pages 362–391, Santa Barbara, CA, USA, 2018. Springer, Heidelberg, Germany. 2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14, 19

GMR01. Y. Gertner, T. Malkin, and O. Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *42nd FOCS*, pages 126–135, Las Vegas, NV, USA, October 14–17, 2001. IEEE Computer Society Press. 2

HO12. B. Hemenway and R. Ostrovsky. Extended-DDH and lossy trapdoor functions. In *PKC 2012*, *LNCS* 7293, pages 627–643, Darmstadt, Germany, May 21–23, 2012. Springer, Heidelberg, Germany. 9

HO13. B. Hemenway and R. Ostrovsky. Building lossy trapdoor functions from lossy encryption. In *ASIACRYPT 2013, Part II*, *LNCS* 8270, pages 241–260, Bengalore, India, December 1–5, 2013. Springer, Heidelberg, Germany. 9

ILL89. R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st ACM STOC*, pages 12–24, Seattle, WA, USA, May 15–17, 1989. ACM Press. 11

Jus72. J. Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on Information Theory*, 18(5):652–656, 1972. 17

KMO10. E. Kiltz, P. Mohassel, and A. O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In *EUROCRYPT 2010*, *LNCS* 6110, pages 673–692, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany. 4, 5, 6, 16

KW18. V. Koppula and B. Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. Cryptology ePrint Archive, Report 2018/847, 2018. https://eprint.iacr.org/2018/847. 9

MPRS12. I. Mironov, O. Pandey, O. Reingold, and G. Segev. Incremental deterministic public-key encryption. In *EUROCRYPT 2012*, *LNCS* 7237, pages 628–644, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. 4

MY10. P. Mol and S. Yilek. Chosen-ciphertext security from slightly lossy trapdoor functions. In *PKC 2010*, *LNCS* 6056, pages 296–311, Paris, France, May 26–28, 2010. Springer, Heidelberg, Germany. 2

PVW08. C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO 2008*, *LNCS* 5157, pages 554–571, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany. 2

PW08. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *40th ACM STOC*, pages 187–196, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press. 2, 3, 5, 6, 7, 8, 9, 23, 24

PW11. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. *SIAM Journal on Computing*, 40(6):1803–1844, 2011. 2, 3, 5, 6, 7, 8, 9, 23, 24

RS09. A. Rosen and G. Segev. Chosen-ciphertext security via correlated products. In *TCC 2009*, *LNCS* 5444, pages 419–436. Springer, Heidelberg, Germany, March 15–17, 2009. 2, 4, 5, 6, 7, 16

RSA78. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signature and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978. 1

Wee12. H. Wee. Dual projective hashing and its applications - lossy trapdoor functions and more. In *EUROCRYPT 2012*, *LNCS* 7237, pages 246–262, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. 2, 8, 9

Wic13. D. Wichs. Barriers in cryptography with weak, correlated and leaky sources. In *ITCS 2013*, pages 111–126, Berkeley, CA, USA, January 9–12, 2013. ACM. 4