On ELFs, Deterministic Encryption, and Correlated-Input Security

Mark Zhandry

Princeton University

Abstract. We construct deterministic public key encryption secure for any *constant* number of *arbitrarily correlated computationally unpredictable* messages. Prior works required either random oracles or non-standard knowledge assumptions. In contrast, our constructions are based on the exponential hardness of DDH, which is plausible in elliptic curve groups. Our central tool is a new *trapdoored extremely lossy function*, which modifies extremely lossy functions by adding a trapdoor.

1 Introduction

The Random Oracle Model [7] is a useful model whereby one models a hash function as a truly random function. Random oracles have many useful properties, such as collision resistance, pseudorandomness, correlation intractability, extractability, and more. Unfortunately, random oracles do not exist in the real world, and some random oracle properties are uninstantiable by concrete hash functions [11]. This has lead to a concerted effort in the community toward constructing hash functions with various strong security properties from standard, well-studied, and widely-accepted assumptions.

Correlated Input Security. In this work, we focus on one particular property satisfied by random oracles, namely correlated input security. Here, the adversary is given $y_i = f(x_i)$ for inputs $x_1, \ldots, x_k$ which may come from highly non-uniform and highly correlated distributions. At the simplest level, we ask that the adversary cannot guess any of the x_i, though stronger requirements such as the pseudorandomness of the y_i are possible. Correlated input security has applications to password hashing and searching on encrypted data [16] and is closely related to related-key security [4]. It is also a crucial security requirement for deterministic public key encryption [3], which is essentially a hash function with a trapdoor.

Correlated input secure functions follow trivially in the random oracle model, and standard-model constructions for specific classes of functions such as low-degree polynomials [16] or "block sources" [8,5,9,14,19] are known. However, there has been little progress toward attaining security for *arbitrary* correlations from standard assumptions, even in the case of just *two* correlated inputs.

This Work. In this work, we construct hash functions and deterministic public key encryption (DPKE) with security for any *constant* number of *arbitrarily* correlated sources. In addition, we only require computational unpredictability for our sources, and our DPKE scheme even achieves CCA security. Our main new technical tool is a new construction of extremely lossy functions (ELFs) [22] that admit a trapdoor. Our construction is secure, assuming that DDH (or more generally k-Lin) is *exponentially* hard to solve. Such an assumption is plausible on elliptic curves.

1.1 Details

We now give an overview of our results and our approach. We start with correlated-input security for one-way functions, and gradually build up to our ultimate goal of deterministic public key encryption.

Correlated Input Secure OWFs. First, we observe that Zhandry's Extremely Lossy Functions (ELFs) [22] already give correlated-input secure one-way functions for any *constant* number of inputs. Recall that an ELF is a variant of a lossy trapdoor function (LTDF), which were introduced by Peikert and Waters [18]. LTDFs are functions with two modes, an injective mode that contains a secret trapdoor for inversion, and a lossy mode that is information-theoretically un-invertible. The security requirement is that these modes are computationally indistinguishable, if you do not know the trapdoor. LTDFs have many applications, including CCA-secure public key encryption [18], deterministic public key encryption [8], and more.

Similarly, and ELF also has two modes, injective and lossy similar to above. However the key difference is that in the lossy mode, the image size is so small that it is actually *polynomial*. Clearly, such a lossy mode can be distinguished from injective by an adversary whose running time is a slightly larger polynomial. So ELFs actually have a spectrum of lossy modes of differing polynomial image sizes, and the exact image size is chosen *based on the adversary* just large enough to fool it. The other main difference between ELFs and LTDFs is that, due to the particulars of Zhandry's construction, the injective mode for ELFs does not contain a trapdoor. Zhandry constructs ELFs based on the *exponential hardness* of the DDH assumption, or more generally exponential k-Lin.

Let f be an injective mode ELF. Consider a source S of d correlated inputs $x_1, \ldots, x_d$ as well as auxiliary information aux. Our goal is to show that, given aux and $f(x_1), \ldots, f(x_d)$, it is computationally infeasible to find any of the x_i. A necessary condition on S is that each x_i are computationally unpredictable given aux alone. Note that we *will* allow sources for which x_i *is* predictable given some of the other x_j. Note that such a source captures the setting where, say, $x_2 = x_1 + 1$, etc.

We now prove that f is one-way for any such computationally unpredictable source. To prove security, we first switch f to be a lossy mode with polynomial image size p. Since d is assumed to be constant, the number of possible value for the vector $f(x_1), \ldots, f(x_d)$ is p^d, also a polynomial. Therefore, this value can be

guessed by the adversary with inverse polynomial probability. As such, if x_i can be guessed given aux and $f(x_1), \ldots, f(x_d)$ with non-negligible probability ϵ, it can also be guessed given just aux with probability at least ϵ/p^d, contradicting the unpredictability of S.

Correlated Input Secure PRGs. Next, we turn to the much harder task of constructing a PRG G for a constant number of correlated inputs. Here, the adversary is given $\mathsf{aux}, y_1, \ldots, y_d$ where either (1) $y_i = G(x_i)$ for all i or (2) y_i is chosen at random in the domain of G. The adversary tries to distinguish the two cases. In order for security to be possible at all, we need to place some minimal restrictions on the source S:

- As in the case of one-wayness, we must require that S is computationally unpredictable
- All the x_i must be distinct, with high probability. Otherwise, the adversary identify the $y_i = G(x_i)$ case by simply testing the equality of the y_i.

In this paper, in order to match notation from Zhandry [22], we will call a function G satisfying indistinguishability a *hardcore function* for computationally unpredictable sources on d inputs.

ELFs are not alone guaranteed to be such hardcore functions, as the outputs are not guaranteed to be random. Instead we build G by starting from Zhandry's hardcore function, which works in the case $d = 1$; that is, for single computationally unpredictable sources. Zhandry's construction is built from ELFs, but requires more machinery to prove pseudorandomness.

The core idea of Zhandry's hardcore function G is the following: first extract *many* Goldreich-Leving hardcore bits, far too many to be simultaneously hardcore. These cannot be output in the clear, as they would allow for trivial inversion. Instead, the bits are scrambled by feeding them through an ELF-based circuit. Zhandry shows (1) that the GL bits can be replaced with random without detection, and (2) if the GL bits are replaced with random, then the output of the circuit is random.

Unfortunately, for correlated sources, the GL bits for different inputs will be correlated: for example if the two inputs differ in a single bit, then if the parity function computing the GL bit is 0 in that position, the two GL bits will be identical. Therefore, the inputs to step (2) in Zhandry's proof may be highly correlated, and his circuit does not guarantee security against correlated inputs.

To mitigate this issue, we carefully modify Zhandry's function G. The idea is, rather than having fixed GL parities, we generate the GL parities as functions of the input itself. Different inputs will hopefully map to independent parities, leading to independent GL bits. We have to be careful, however, in order to avoid any circularities in the analysis, since we need the GL parities to be (pseudo)random and independent (in order to apply the GL theorem), but generating such random independent parities already seems to require extracting pseudorandom strings for arbitrarily correlated sources, leaving us back where we started.

Our construction works as follows: we have another ELF instance, which is applied to the input x, resulting in an value w. Then we apply a d-wise independent function R to w to generate the actual parities. Zhandry shows that this composition of an ELF and a d-wise independent function is collision-resistant for $d \geq 2$, meaning the d different x_i will indeed map to distinct parities, and in fact distinct w_i. Next, in the lossy mode for the ELF, there are only a polynomial number of w; since d is constant, there are also a polynomial number of possible d-tuples of $(w_1, \ldots, w_d)$. Therefore, we can actually *guess* the $(w_1, \ldots, w_d)$ vector that will result from applying the ELF to the x_i with inverse-polynomial probability. Next, since R is d-wise independent, we can take d independent sets of GL parities and program R to output these parities on the corresponding d values of w. This is not quite enough to show that the GL parities are themselves pseudorandom for correlated sources (since we only successfully program with inverse-polynomial probability), but with a careful proof we show that it is sufficient to prove the pseudorandomness of the overall construction.

Deterministic Public Key Encryption. Next, we turn to constructing deterministic public key encryption (DPKE). A DPKE protocol consists of 3 algorithms, $(\mathsf{DPKE.Gen}, \mathsf{DPKE.Enc}, \mathsf{DPKE.Dec})$. $\mathsf{DPKE.Gen}$ creates a secret/public key pair sk, pk. $\mathsf{DPKE.Enc}$ is a deterministic procedure that uses the public key pk to scramble a message m, arriving at a ciphertext c. $\mathsf{DPKE.Dec}$ is also deterministic, and maps the ciphertext c back to m.

We first consider security in the single-input setting; we note that it was previously open to construction DPKE for even a single arbitrary computationally unpredictable source. The canonical way to build DPKE [3] is to use an ordinary *randomized* public key encryption scheme with CPA security. The idea is to hash the message m using a hash function H, and use $H(m)$ as the randomness r: $\mathsf{DPKE.Enc}(\mathsf{pk}, m) = \mathsf{DPKE.Enc}(\mathsf{pk}, m; H(r))$ where $\mathsf{DPKE.Enc}$ is the randomized PKE encryption algorithm. In the random oracle model for H, Bellare, Boldyreva and O'Neill [3] show that this scheme obtains the strongest possible notion of security. One may hope that some ELF-based hash function H might be sufficient.

Unfortunately, Brzuska, Farshim and Mittelbach [10] give strong evidence that this scheme cannot be proven secure in the standard model, even under very strong assumptions. In particular, they devise a public key encryption scheme $\mathsf{PKE.Enc}$ such that, for any concrete hash function H, $\mathsf{DPKE.Enc}$ will be insecure. Their construction uses indistinguishability obfuscation [2] (iO), which is currently one of the more speculative tools used in cryptography. Nonetheless, in order to give a standard model construction of DPKE, one must either deviate from the scheme above, or else prove conclusively that iO does not exist.

On the other hand, lossy trapdoor functions have proven useful for building DPKE in the standard model (e.g. [8,9]). One limitation of these techniques, however, is that since the image size of a LTDF is always at least sub-exponential, constructions based on LTDFs tend to require high min-entropy/computational unpredictability requirements.

Our first construction. We start by abstracting the constructions of Brakerski and Segev [9]. They construct DPKE for sub-exponentially unpredictable sources by essentially analyzing specific constructions of Lossy Trapdoor Functions (LTDFs), and showing that they satisfy the desired security experiment.

Our first construction abstracts their construction to work with arbitrary LTDFs. Our construction is the following, based on a semantically secure public key encryption scheme $\mathsf{PKE.Enc}$, a special kind of pseudorandom generator G, and a LTDF f generated in the injective mode:

$$\mathsf{DPKE.Enc}(\mathsf{pk}, m) = \mathsf{PKE.Enc}(\mathsf{pk}, f(m); G(m))$$

To prove security, we first switch to f being in the lossy mode. Now, notice that if m can be predicted with probability p, then it can still be predicted with probability p/r even given $f(m)$, by simply guessing the value of $f(m)$, which will be correct with probability $1/r$. In particular, if p is sub-exponentially small and r is sub-exponential, then p/r is also sub-exponential. Any LTDF can be set to have a sub-exponential-sized lossy mode by adjusting the security parameter accordingly.

Next, we observe that if G is hardcore for sub-exponentially unpredictable sources, then $G(m)$ will be pseudorandom given $f(m)$. Such a G can be built by extracting a sufficiently small polynomial-number of Goldreich-Levin [15] bits, and then expanding using a standard PRG.

At this point, we can replace $G(m)$ with a random bitstring, and then rely on the semantic security of $\mathsf{PKE.Enc}$ to show security, completing the security proof.

But what about *arbitrary* computationally unpredictable sources, which may not be sub-exponentially secure? Intuitively, all we need is that (1) r can be taken to be an arbitrarily small super-polynomial, and (2) that G is secure for arbitrary unpredictable sources, instead of just sub-exponential sources. We then recall that Zhandry's [22] construction of G already satisfies (2), and that ELF's themselves *almost* satisfy (1). Unfortunately, the resulting scheme is not an encryption scheme: Zhandry's ELFs do not have a trapdoor in the injective mode, meaning there is no way to decrypt.

Therefore, we propose the notion of a trapdoor ELFs, which combines the functionality of ELFs and LTDFs by allowing for both a polynomial image size *and* a trapdoor. Using a trapdoor ELF, the above construction becomes a secure DPKE scheme for any computationally unpredictable source. For now we will simply assume such trapdoor ELFs; we discuss constructing such functions below.

CCA Security. Next we turn to achieving CCA security for DPKE. CCA security has received comparatively less attention in the deterministic setting, though some standard-model constructions are known [8,19,17]. In particular, we are not aware of any constructions for computationally unpredictable sources, sub-exponentially hard or otherwise.

We observe that by combining techniques for building CCA-secure encryption from LTDFs [18,8] with our abstraction of Brakerski and Segev [9], we can achieve CCA security for sub-exponentially hard sources. The idea is to use all-but-one

LTDFs, a generalization of LTDFs where the function f has many branches. In the injective mode, each branch is injective. In the lossy mode, a single branch is lossy, and the inverse function works for all other modes. The adversary cannot tell injective from lossy, even if it knowns the branch b^*. Peikert and Waters [18] show how to generically construct such ABO LTDFs from any LTDF.

First, we modify the definition to require that indistinguishability from injective and lossy holds even if the adversary can make inversion queries on all branches other than b^*. The generic construction from standard LTDFs satisfies this stronger notion.

Then, our CCA-secure construction can be seen as combining our construction above with the construction of [8]. We encrypt using the algorithm $\mathsf{DPKE.Enc}(\mathsf{pk}, m) = (\ b = G_0(m), \mathsf{PKE.Enc}(\mathsf{pk}, f(b, m); G_1(m))\ )$ where G_0, G_1 are strong pseudorandom generators, and $f(b, m)$ is the ABO LTDF evaluation on branch b. Here, we require $\mathsf{PKE.Enc}$ to be a CCA-secure PKE scheme.

Intuitively, G_0 determines the branch, and if it is injective, then each message has its own branch. Once the branch is fixed, the rest of the scheme basically becomes our basic scheme from above. The challenge ciphertext will be set to be the lossy branch, which can be proven to hide the message following the same proof as our basic scheme. We will need to simulate CCA queries, which can be handled by using the CCA-security of $\mathsf{PKE.Enc}$ and the security of f under inversion queries.

Using standard LTDFs, we thus get the first CCA-secure scheme for sub-exponentially hard computationally unpredictable sources

Turning to the setting of arbitrary unpredictable sources, we need to replace the ABO LTDF with an ABO trapdoor ELF, which works. Unfortunately, as discussed below, the generic construction of ABO LTDF in [18] does not apply to trapdoor ELFs, so we need a different approach to construct an ABO trapdoor ELF. Our approach is outlined below when we discuss our ELF constructions.

Correlated Inputs. Next, we turn to constructing DPKE for correlated inputs. Here, we require essentially the same security notion as for hardcore functions; the only difference is in functionality, since there is a trapdoor for inversion.

In the case of CPA security, security trivially follows if we replace G with our hardcore function for correlated inputs. We therefore easily get the first DPKE scheme secure for a constant number of correlated sources. We also note that if the source is sub-exponentially unpredictable, our scheme can be based on standard LDTFs.

We can similarly extend this idea to get CCA security. Except here, we will need a trapdoor ELF with several lossy branches, one for each challenge ciphertext.

Constructing Trapdoor ELFs. Finally, we turn to actually constructing trapdoor ELFs. Our trapdoor ELFs will be based on Zhandry's ELFs, which are in turn based on constructions of LTDFs [13]. But unfortunately, Zhandry's ELFs lose the trapdoor from the LTDFs. Here, we show how to resurrect the trapdoor.

Zhandry's construction basically iterates Freeman et al.'s [13] LTDF at many security levels. Freeman et al.'s construction expands the inputs by a modest factor. Thus, Zhandry needs to compress the outputs of each iteration in order for the size to not grow exponentially. Unfortunately, this compression results in the trapdoor being lost, since it is un-invertible.

Instead, we opt to avoid compression by more carefully choosing the security parameters being iterated. Zhandry chooses powers of 2 from 2 up to the global security parameter. Instead, we choose double exponentials 2^{2^i}. We still cannot go all the way to the global security parameter, but we show that we can go high enough in order to capture any polynomial. Thus, we obtain ELFs that admit a trapdoor.

For our application to CCA-security, we need to introduce branches into our trapdoor ELFs. Unfortunately, the approach of Peikert and Waters [18] is insufficient for our purposes. In particular, they introduce branching by applying many different LTDFs in parallel to the same input, outputting all images. The overall image size is then roughly the product of the image sizes of each underlying LTDF. The branch specifies which subsets of LTDFs are applied; the LTDFs corresponding to the lossy branch are all set to be lossy. In this way, the lossy branch will indeed be lossy. On the other hand, any other branch will have at least one LTDF which is injective, meaning the overall function is injective. Unfortunately for us, this approach results in an exponential blowup in the size of the image space for the lossy branch, even if the original image size was polynomial. Hence, applying this transformation to an ELF would not result in an ABO ELF.

Instead, we opt for a direct construction though still based on Freeman et al.'s scheme. Recall Freeman et al.'s scheme: the function f^{-1} is specified by an $n \times n$ matrix $\mathbf{A}$ over $\mathbb{Z}_q$, and the function f is specified by $\mathbf{A}$, but encoded in the exponent of a cryptographic group over order q: $g^{\mathbf{A}}$. The function f takes an input $\mathbf{x} \in \{0,1\}^n$, and maps it to $g^{\mathbf{A}\cdot\mathbf{x}}$ by carrying out appropriate group operations on $g^{\mathbf{A}}$. The inverse function f^{-1} uses $\mathbf{A}^{-1}$ to recover $g^{\mathbf{x}}$ from $g^{\mathbf{A}\cdot\mathbf{x}}$, and then solves for $\mathbf{x}$, which is efficient since $\mathbf{x}$ is 0/1.

In the lossy mode, $\mathbf{A}$ is set to be a matrix of rank 1. By DDH, this is indistinguishable from full rank when just given $g^{\mathbf{A}}$. On the other hand, now the image size of f is only q. By setting $2^n \gg q$, this function will now be lossy.

We now give a direct construction of an ABO trapdoor ELF. Our idea is to make the matrices tall, say $2n$ rows and n columns. Note that *any* left inverse of $\mathbf{A}$ will work for inverting the function, and there are many.

Our actual construction is the following. For branches in $\{0,1\}^a$, f^{-1} will be specified by $2a+1$ matrices $\mathbf{B}, \mathbf{A}_{i,t}$ for $i \in [a], t \in \{0,1\}$. The description of f will simply be the corresponding encoded values of $\mathbf{B}, \mathbf{A}_{i,t}$. The branch $b \in \{0,1\}^a$ corresponds to the matrix $\mathbf{A}_b = \mathbf{B} + \sum_i \mathbf{A}_{i,b_i}$.

For a lossy mode with branch b, we set $\mathbf{A}_b$ to be rank 1. Then we choose $\mathbf{A}_{i,t}$ at random and set $\mathbf{B} = \mathbf{A}_b - \sum_i \mathbf{A}_{i,b_i}$.

We would now like to prove security. For a given branch b^*, suppose an adversary can distinguish the injective mode from the mode where b^* is lossy. We

now show how to use such an adversary to distinguish $g^{\mathbf{C}}$ for a full-rank $n \times n$ $\mathbf{C}$ from a random rank-1 $\mathbf{C}$.

First, we will set $\mathbf{A}_{b^*}$ to be the matrix $\mathbf{C}$, except with n more rows appended, all of which are zero. We can easily construct $g^{\mathbf{A}_{b^*}}$ from $g^{\mathbf{C}}$ without knowing $\mathbf{C}$. Then we choose random $\mathbf{A}_{i,t}$. Finally, we set $\mathbf{B} = \mathbf{A}_{b^*} - \sum_i \mathbf{A}_{i,b_i}$. We can easily construct $g^{\mathbf{B}}$ given $g^{\mathbf{C}}$, again without knowing $\mathbf{C}$.

Now notice that for each branch b, we know the bottom n rows of $\mathbf{A}_b$, and moreover for $b \neq b^*$ they are full rank. Therefore, we can invert on any branch other than b^*, allowing us to simulate the adversary's queries.

Unfortunately, the distribution simulated is not indistinguishable from the correct distribution. After all, $\mathbf{A}_{b^*}$ is all zeros on the bottom n rows, which is easily detectable by the adversary. In order to simulate the correct distribution, we actually left-multiply all the matrices $\mathbf{B}, \mathbf{A}_{i,t}$ by a random matrix $\mathbf{R} \in \mathbb{Z}_q^{2n \times 2n}$. This can easily be done in the exponent. Moreover, now in the case where $\mathbf{C}$ is random, the matrices $\mathbf{B}, \mathbf{A}_{i,t}$ are actually random. On the other hand if $\mathbf{C}$ is rank 1, we correctly simulate the case where b^* is lossy.

Our construction above can easily be extended to multiple lossy branches by iterating the construction several times, one for each branch that needs to be lossy. Then, we notice that we actually achieve a polynomial image size by setting q to be a polynomial, and then relying on the exponential hardness of DDH to prove indisitnguishability. Thus, we achieve trapdoor ELFs with multiple lossy branches, as needed for our construction.

1.2 Discussion

Of course, one way to achieve a hash function with security for correlated inputs — or more generally any security property — is to simply make the "tautological" assumption that a given hash function such as SHA has the property. Assuming the hash function is well designed, such an assumption may seem plausible. In fact, for practical reasons this is may be the preferred approach.

However, in light of the impossibility of instantiating random oracles in general [11], it is a priori unclear which hash function properties are achievable in the standard model. It could be, for example, that certain correlations amongst inputs will always be trivially insecure, even for the best-designed hash functions. The *only* way to gain confidence that a particular hash function property is plausible at all is to give a construction provably satisfying the property under well-studied and widely accepted computational assumptions. Our correlated-input secure PRG G does exactly this.

On Exponential Hardness. Our constructions rely on the exponential hardness of DDH, which is plausible in elliptic curve groups based on the current state of knowledge. Elliptic curves have been studied for some time, and to date no attack has been found that violates the exponential hardness in general elliptic curves.

In fact, exponential hardness is exactly what makes elliptic curves desirable for practical cryptographic applications today. DDH over finite fields is solveable in

subexponential time, meaning parameters must be set much larger to block attacks. This leads to much less efficient schemes. Some of the most efficient protocols in use today rely on elliptic curves, precisely because we can set parameters aggressively and still remain secure. Thus, the exponential hardness of DDH in elliptic curve groups is widely assumed for real-world schemes.

We also remark that, as explained by Zhandry [22], polynomial-time and even sub-exponential-time hardness are insufficient for one-way functions secure for arbitrary min-entropy sources, which in particular are implied by our correlated-input secure constructions. Therefore, some sort of extremely strong hardness is inherent in our applications.

Concretely, security for arbitrary min-entropy sources implies the following: for any super-logarithmic function $t(n)$, there is a problem in NP that (1) only requires $t(n)$ bits of non-determinism, but (2) is still not contained in P. Put another way, the problem can be brute-forced in very slightly super-polynomial time, but is not solvable by *any* algorithm in polynomial time, showing that brute-force is essentially optimal. This can be seen as a scaled-down version of the exponential time hypothesis. Thus, while exponential hardness may not be required for the applications, a scaled-down version of exponential hardness *is* required.

Common Random String. Our constructions are based on Zhandry's ELFs, which require a common random string (crs); this crs is just the description of the injective-mode function. Thus our hardcore functions require a crs, and moreover, we only obtain security if the crs is sampled independently of the inputs. A natural question is whether this is required. Indeed, the following simple argument shows that pseudorandomness for even a *single information-theoretically unpredictable* source is impossible without a crs. After all, for a fixed function G, let S sample a random input x conditioned on the first bit of $G(x)$ being 0. Then the first bit of $G(x)$ will always be zero, whereas the first bit of a random string will only be zero half the time. This argument also easily extends to the setting of a crs, but where the sampler *depends* on a crs. It also extends for security for DPKE schemes where the messages depend on the public key, as noted in [19].

Even if we restrict to inputs that are statistically close to uniform, but allow two inputs to be slightly correlated, a crs is still required for pseudorandomness. Indeed, for a function G that outputs n-bit strings, consider the following sampler: choose two inputs x_0, x_1 at random, conditioned on the first bit of $G(x_0) \oplus G(x_1)$ being 0.

In the case of one-wayness, the above does not quite apply (since $G(x)$ may still hide x), but we can show that one-wayness without a crs is impossible for any super-constant number of correlated inputs. Basically, for d inputs, the sampler S will choose a random $(d-1)\log\lambda$-bit string x_1, which has super-logarithmic min-entropy since d is super-constant. Then it will divide x into $d-1$ blocks of $\log\lambda$ bits $z_2, \ldots, z_d$. It will then sample random $x_2, \ldots, x_d$ such that the first $\log\lambda$ bits of $G(x_i)$ are equal to z_i (which requires $O(\lambda)$ evaluations of G). Finally, it outputs $x_1, \ldots, x_d$. Given the outputs $y_1, \ldots, y_d$, it is easy to reconstruct x_1.

Of course, we only achieve security for a constant number of correlated inputs *with* a crs, so this leaves open the interesting problem of constructing a one-way function for a constant number of correlated inputs *without* using a crs.

Barriers to Correlated-Input Security. Even with a crs, correlated-input security has been difficult to achieve. The following informal argument from Wichs [21] gives some indication why this is the case. Let P_1, P_2 be two functions. Consider correlated x_1, x_2 sampled as $x_1 = P_1(r), x_2 = P_2(r)$, *for the same choice of random* r. Now, a reduction showing correlated-input security would need to transform an attacker A for the correlated inputs into an algorithm B for some presumably hard problem. But it seems that B needs to some how feed into A a valid input $G(x_1), G(x_2)$, and then use A's attack in order to solve it's problem. But the only obvious way to generate a valid input for general P_1, P_2 is to choose a random r and set $x_1 = P_1(r), x_2 = P_2(r)$. But then B already knows what A would do, making A's attack useless.

The standard way (e.g. [1]) to get around this argument is to use G that are lossy, and this is the approach we use, exploiting the two modes of the ELF. Our results show that it is possible to attain security for a constant number of inputs.

What about larger numbers of correlated inputs? Wichs [21] shows that proofs relative to polynomial-time falsifiable assumptions that make *black-box* use of the adversary are impossible for any *super-logarithmic* number of correlated messages. Note that the impossibility does *not* apply to our results for three reasons:

- Our reduction requires knowing the adversary's success probability and running time, and is therefore very slightly non-black box. In the language of [12], our reduction is "non-uniform"
- We require exponential hardness, not polynomial-time hardness
- We only achieve a constant number of correlated messages.

Nonetheless, Wichs impossibility represents a barrier to significantly improving our results.

Deterministic Public Key Encryption. Deterministic public key encryption can be thought of as an injective hash function that also has a trapdoor. As a result, the definitions of security for DPKE are related to strong security notions for hash functions such as correlated-input security. We note that [6] construct correlated-input secure DPKE for an *arbitrary* number of correlated min-entropy sources. Their underlying building blocks are LTDFs and *universal computational extractors* (UCE's). Note that UCE's are a strong "uber" type assumption on hash functions that includes many different security properties, including correlated-input security. Therefore, the main difficulty in their work is showing how to take a hash function that already attains the security notion they want (and then some) and then building from it a function that also has a trapdoor.

Our correlated-input secure hash function is likely not a UCE. In particular, in light of Wich's impossibility results discussed above, we don't expect to be able to prove that our construction is correlated-input secure for a large number of inputs. More we do not expect to be able to prove all UCE security properties for

our assumption. Therefore, we cannot simply plug our hash function construction into [6] to get a DPKE scheme.

2 Preliminaries

Definition 1. *Consider a distribution D on pairs (x, aux), indexed by the security parameter λ. We say that D is* computationally unpredictable *if, for any probabilistic polynomial time adversary A, there is a negligible function ϵ such that*

$$\Pr[A(\mathsf{aux}) = x : (x, \mathsf{aux}) \leftarrow D(\lambda)] < \epsilon(\lambda)$$

In other words, A cannot efficient guess x given aux

Lemma 1. *Let D be a source of tuples (x, aux, z) such that (x, aux) is computationally unpredictable. Let $\mathcal{F}$ be a distribution over functions f with the following property. $f(\mathsf{aux}, x, z)$ is function such that, for any aux, $f(\mathsf{aux}, \cdot, z)$ has polynomial image size, and that it is possible to efficiently compute the polynomial-sized image. Then D' which samples $(x, \mathsf{aux}' = (\mathsf{aux}, f, f(\mathsf{aux}, x, z)))$ is computationally unpredictable.*

Proof. If there is an A adversary for D', we can simply make a random guess for the value of $f(\mathsf{aux}, x, z)$, which will be right with inverse polynomial probability. In this case, we correctly simulate the view of A, meaning A outputs x with non-negligible probability. Overall, we break the computational unpredictability of x with inverse polynomial probability □

We will also consider a notion of computationally unpredictable sources on multiple correlated inputs:

Definition 2. *Consider a distribution D on tuples $(x_1 \ldots, x_d, \mathsf{aux})$, indexed by the security parameter λ. We say that D is* computationally unpredictable *if the following hold:*

- *For any $i \neq j$, $\Pr[x_i = x_j]$ is negligible.*
- *For any probabilistic polynomial time adversary A, there is a negligible function ϵ such that*

$$\Pr[A(\mathsf{aux}) \in \{x_1, \ldots, x_d\} : (x_1, \ldots, x_d, \mathsf{aux}) \leftarrow D(\lambda)] < \epsilon(\lambda)$$

In other words, each distribution (x_i, aux) is computationally unpredictable.

Notice we do *not* require x_i to be unpredictable given $x_j, j \neq i$. As such, distributions such as $x, x+1, x+2, \mathsf{aux} = \emptyset$ are considered unpredictable.

We now consider hardcore functions:

Definition 3. *Let $\mathcal{G}$ be a sampling procedure for deterministic functions G on $n = n(\lambda)$ bits with $m = m(\lambda)$ bit outputs. We say that $\mathcal{G}$ is* hardcore for any computationally unpredictable source *if for any computationally unpredictable*

source D for $x \in \{0,1\}^n$, and any adversary A, there is a negligible function ϵ such that:

$$|\Pr[A(G, G(x), \mathsf{aux}) = 1] - \Pr[A(G, R, \mathsf{aux})]| < \epsilon(\lambda)$$

where $G \leftarrow \mathcal{G}$, $(x, \mathsf{aux}) \leftarrow D(\lambda)$ and R is random in $\{0,1\}^m$. In other words, $G(x)$ is pseudorandom even given aux.

Definition 4. *Let $\mathcal{G}$ be a sampling procedure for deterministic functions G on $n = n(\lambda)$ bits with $m = m(\lambda)$ bit outputs. We say that $\mathcal{G}$ is* hardcore for any computationally unpredictable source *over d-inputs if for any computationally unpredictable source D for d inputs $x_1 \ldots, x_d \in \{0,1\}^n$, and any adversary A, there is a negligible function ϵ such that:*

$$|\Pr[A(G, G(x_1), \ldots, G(x_d), \mathsf{aux}) = 1] - \Pr[A(G, R_1, \ldots, R_d, \mathsf{aux})]| < \epsilon(\lambda)$$

where $G \leftarrow \mathcal{G}$, $(x_1, \ldots, x_d, \mathsf{aux}) \leftarrow D(\lambda)$ and $R_1 \ldots, R_d$ are random in $\{0,1\}^m$. In other words, $G(x)$ is pseudorandom even given aux *and the correlated inputs.*

2.1 Deterministic Public Key Encryption

A deterministic public key encryption scheme is a tuple of efficient algorithms $(\mathsf{DPKE.Gen}, \mathsf{DPKE.Enc}, \mathsf{DPKE.Dec})$, where $\mathsf{DPKE.Enc}, \mathsf{DPKE.Dec}$ are deterministic maps between messages and ciphertexts, and $\mathsf{DPKE.Gen}$ is randomized procedure for producing secret and public key pairs.

For security, we consider several possible notions. Security for arbitrary computational sources means that $(\mathsf{pk}, c^* = \mathsf{DPKE.Enc}(\mathsf{pk}, m), \mathsf{aux})$ is computationally indistinguishable from $(\mathsf{pk}, c^* = \mathsf{DPKE.Enc}(\mathsf{pk}, R), \mathsf{aux})$, where (m, aux) is sampled from an arbitrary computationally unpredictable source and R is uniformly random, and $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{DPKE.Gen}(\lambda)$. CCA security means the same holds even if the adversary can later ask for decryption queries on ciphertexts other that c^*. Security for arbitrary correlated sources means that $(\mathsf{pk}, \mathsf{DPKE.Enc}(\mathsf{pk}, m_1), \ldots, \mathsf{DPKE.Enc}(\mathsf{pk}, m_1), \mathsf{aux})$ is indistinguishable from $(\mathsf{pk}, \mathsf{DPKE.Enc}(\mathsf{pk}, R_1), \ldots, \mathsf{DPKE.Enc}(\mathsf{pk}, R_d), \mathsf{aux})$ for arbitrary computationally unpredictable sources on d inputs.

2.2 ELFs

We recall the basic definition of Extremely Lossy Functions (ELFs) from Zhandry. We slightly change notation, but the definition is equivalent.

A Lossy Trapdoor Function, or LTDF [18], is a function family with two modes: an injective mode where the function is injective and there is a trapdoor for inversion, and a lossy mode where the image size of the function is much smaller than the domain. The security requirement is that no polynomial-time adversary can distinguish the two modes. An Extremely Lossy Function, or ELF [22], is a related notion without a trapdoor in the injective mode, but with a more powerful lossy mode. In particular, in the lossy mode the image size can

be taken to be a polynomial r. One fixed polynomial r is insufficient (since then the lossy mode could easily be distinguished from injective), but instead, the polynomial r is tuned based on the adversary in question to be just large enough to fool the adversary.

Definition 5 (Zhandry [22]). *An ELF consists of two algorithms* $\mathsf{ELF.GenInj}$ *and* $\mathsf{ELF.GenLossy}$*, as well as a function* $N = N(M)$ *such that* $\log N$ *is polynomial in* $\log M$*.* $\mathsf{ELF.GenInj}$ *takes as input an integer* M*, and outputs the description of a function* $f : [M] \to [N]$ *such that:*

- *f is computable in time polynomial in the bit-length of their input, namely* $\log M$*.*
- *With overwhelming probability (in* $\log M$*), f is injective*

$\mathsf{ELF.GenLossy}$ *on the other hand takes as input integers* M *and* $r \in [M]$*. It outputs the description of a function* $f : [M] \to [N]$ *such that:*

- *For all* $r \in [M]$*,* $|f([M])| \leq r$ *with overwhelming probability. That is, the function f has image size at most r.*
- *For any polynomial p and inverse polynomial function δ (in* $\log M$*), there is a polynomial q such that: for any adversary $\mathcal{A}$ running in time at most p, and any* $r \in [q(\log M), M]$*, we have that*

$$|\Pr[\mathcal{A}(f) = 1 : f \leftarrow \mathsf{ELF.GenInj}(M)] - \Pr[\mathcal{A}(f) = 1 : f \leftarrow \mathsf{ELF.GenLossy}(M, r)| < \delta$$

In other words, no polynomial-time adversary $\mathcal{A}$ can distinguish an injective f from an f with polynomial image size.

3 Correlated-Input Hardcore Functions

In this section, we build our correlated-input hardcore function. First, we recall Zhandry's [22] construction of hardcore functions for arbitrarily uncorrelated sources. The following description is taken essentially verbatim from Zhandry.

Construction 1 *Let q be the input length and m be the output length. Let λ be a security parameter. We will consider inputs x as q-dimensional vectors* $\mathbf{x} \in \mathbb{F}_2^q$*. Let* ELF *be an ELF. Let* $M = 2^{m+\lambda+1}$*, and let n be the bit-length of the ELF on input $m+1$. Set* $N = 2^n$*. Let ℓ be some polynomial in m, λ to be determined later. First, we will construct a function H' as follows.*

Choose random $f_1, \ldots, f_\ell \leftarrow \mathsf{ELF.GenInj}(M)$ *where* $f_i : [M] \to [N]$*, and let* $h_1, \ldots, h_{\ell-1} : [N] \to [M/2] = [2^{m+\lambda}]$ *and* $h_\ell : [N] \to [2^m]$ *be sampled from pairwise independent and uniform function families. Define* $\mathbf{f} = \{f_1, \ldots, f_\ell\}$ *and* $\mathbf{h} = \{h_1, \ldots, h_\ell\}$*. Define* $H'_i : \{0,1\}^i \to [M/2]$ *(and* $H'_\ell : \{0,1\}^\ell \to [2^m]$*) as follows:*

- $H'_0() = 1 \in [2^{m+\lambda}]$

– $H'_i(\mathbf{b}_{[1,i-1]}, b_i)$: *compute* $y_i = H'_{i-1}(\mathbf{b}_{[1,i-1]})$, $z_i \leftarrow f_i(y_i||b_i)$, *and output* $y_{i+1} \leftarrow h_i(z_i)$

Then we set $H' = H'_\ell$. *Then to define* H, *choose a random matrix* $\mathbf{R} \in \mathbb{F}_2^{\ell \times q}$. *The description of* H *consists of* $\mathbf{f}, \mathbf{h}, \mathbf{R}$. *Then set* $H(x) = H'(\mathbf{R} \cdot \mathbf{x})$. *A diagram of* H *is given in Figure 1.*

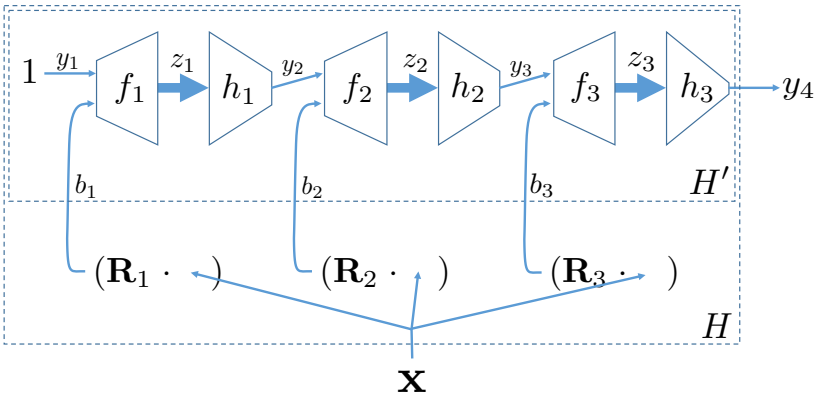

Fig. 1: An example taken from Zhandry [22] for $\ell = 3$. Notice that each iteration is identical, except for the final iteration, where h_ℓ has a smaller output.

Our Construction. We will modify Zhandry's construction as follows. Sample $\mathbf{f}, \mathbf{h}$ as in Construction 1. Then define the function $H_{\mathbf{R}}(\mathbf{x})$ to be the function H using Goldreich-Levin parities $\mathbf{R}$.

Our modification will be to generate $\mathbf{R}$ as a function of $\mathbf{x}$, and then apply $H_{\mathbf{R}}(\mathbf{x})$. In particular, we will set $\mathbf{R} = u(v(\mathbf{x}))$ where $v \leftarrow \mathsf{ELF.GenInj}(M)$ and u is a d-wise independent function. Actually, we need a stronger property of u: that each row of $\mathbf{R}$ is specified by an independent d-wise independent function u_i.

Theorem 2. *If* ELF *is a secure ELF, then* $H_{u(v(\mathbf{x}))}(\mathbf{x}) = H'(u(v(\mathbf{x})) \cdot \mathbf{x})$ *is a hardcore function for computationally unpredictable sources on* d *inputs, for any constant* d.

Proof. First, we recall some basic facts proved by Zhandry:

Claim. If $\ell \geq m + \lambda$, and if $\mathbf{b}$ is drawn uniformly at random, then $(H', H'(\mathbf{b}))$ is statistically close to (H', R) where R is uniformly random in $[2^m]$.

Therefore, given a source D which samples messages $m_1, \ldots, m_d$ and auxiliary information aux, it is sufficient to prove the following are indistinguishable: $(\ \mathbf{f}, \mathbf{h}, u, v, \mathsf{aux}, \{H'(u(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i\ )$ and $(\ \mathbf{f}, \mathbf{h}, u, v, \mathsf{aux}, \{H'(\mathbf{b}_i)\}_i\ )$ for uniformly random $\mathbf{b}_i$.

Our proof will follow the same high-level idea as in Zhandry, but make adjustments along the way in order to prove security for correlated sources. Let

$\mathcal{A}$ be an adversary with non-negligible advantage ϵ in distinguishing the two cases. We will assume it always checks that the images $v(\mathbf{x}_i)$ are all distinct and rejects if they are; by the property of the source D and the injectivity of v, this check will never trigger if sampled as above. Nonetheless, if the check triggers, we assume A outputs a random bit and aborts.

Let $\mathbf{R}_i = u(v(\mathbf{x}_i))$. Define $\mathbf{b}_i^{(j)}$ so that the first j bits of $\mathbf{b}_i^{(j)}$ are equal to the first j bits of $\mathbf{R}_i \cdot \mathbf{x}_i$, and the last $\ell - i$ bits are uniformly random and independent of $\mathbf{x}_1, \ldots, \mathbf{x}_d$.

We now define a sequence of hybrids. In **Hybrid** j, $\mathcal{A}$ is given the distribution $(\ \mathbf{f}, \mathbf{h}, u, v, \mathsf{aux}, \{H'(\mathbf{b}_i^{(j)})\}_i\ )$. Then $\mathcal{A}$ distinguishes **Hybrid 0** from **Hybrid** ℓ with probability ϵ. Now we choose an j at random from $[\ell]$. The adversary distinguishes **Hybrid** $j-1$ from **Hybird** j with expected advantage at least ϵ/ℓ. Next, observe that since bits $j+1$ through t are random in either case, they can be simulated independently of the challenge. Moreover, $H'(\mathbf{b})$ can be computed given $H'_{j-1}(\mathbf{b}_{[j-1]})$, the bit b_j (be it random or equal to $\mathbf{R} \cdot \mathbf{x}$), and the random $b_{j+1}, \ldots, b_\ell$. Also, the d-wise independent functions $u_{j+1}, \ldots, u_\ell$ are never evaluated on the $\mathbf{x}_i$, so they can be simulated as well. Let $u_{[j]}(x)$ denote the output $(u_1(x), \ldots, u_j(x))$.

Thus, we can construct an adversary $\mathcal{A}'$ that distinguishes the following distributions:

$$(\ j, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_j, v, \mathsf{aux}, \{H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i), u_j(v(\mathbf{x}_i)) \cdot \mathbf{x}_i\}_i\ ) \text{ and}$$
$$(\ j, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_j, v, \mathsf{aux}, \{H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i), b_i\}_i\ )$$

with advantage ϵ/ℓ, where j is chosen randomly in $[\ell]$, where b_i are random bits.

Next, notice that $\epsilon/5\ell$ is non-negligible, meaning there is an inverse polynomial δ such that $\epsilon/5\ell \geq \delta$ infinitely often. Then, there is a polynomial r such $\mathcal{A}'$ cannot distinguish f_i generated as $\mathsf{ELF.GenLossy}(M, r)$ from the honest f_i generated from $\mathsf{ELF.GenInj}(M)$, except with probability at most δ. Similarly we'll generate v by $\mathsf{ELF.GenLossy}(M, r)$.

This means, if we generate $f_i, v \leftarrow \mathsf{ELF.GenLossy}(M, r)$, we have that $\mathcal{A}'$ still distinguishes the distributions

$$(\ j, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_j, v, \mathsf{aux}, \{H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i), u_j(v(\mathbf{x}_i)) \cdot \mathbf{x}_i\}_i\ ) \text{ and}$$
$$(\ j, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_j, v, \mathsf{aux}, \{H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i), b_i\}_i\ )$$

with advantage $\epsilon' = \epsilon/\ell - 4\delta$.

Next, we define new hybrids $J_0, \ldots, J_d$, where J_k is the distribution:

$$(\ j, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_j, v, \mathsf{aux}, \{H'_{i-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i), q_i\}_i\ )$$

where $q_i = u_j(v(\mathbf{x}_i)) \cdot \mathbf{x}_i$ for $i \leq k$ and q_i is uniformly random for $i > k$. Notice that J_0 and J_d ar the two distributions distinguished with probability ϵ'. Therefore, for a random $k \in [d]$, the expected distinguishing advantage between J_{i-1} and J_i is ϵ'/d. Thus, $\mathcal{A}'$ can be used to construct an adversary $\mathcal{A}''$ that distinguishes the two distributions:

$$\begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, \{u_i\}_i, v, \mathsf{aux}, \\ \{H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_{i\in[d]}, \{u_j(v(\mathbf{x}_i)) \cdot \mathbf{x}_i\}_{i<k}, u_j(v(\mathbf{x}_k)) \cdot \mathbf{x}_k \end{pmatrix} \text{ and}$$
$$\begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, \{u_i\}_i, v, \mathsf{aux}, \\ \{H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_{i\in[d]}, \{u_j(v(\mathbf{x}_i)) \cdot \mathbf{x}_i\}_{i<k}, b_k \end{pmatrix}$$

with advantage $\epsilon'/4$. Next, we devise an adversary $\mathcal{A}'''$ which distinguishes

$$\begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, \{u_i\}_i, v, \mathsf{aux}, \\ \{v(\mathbf{x}_i), H'_{i-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i, \{\mathbf{r}_i \cdot \mathbf{x}_i\}_{i<k}, \mathbf{r}_k \cdot \mathbf{x}_k \end{pmatrix} \text{ and}$$
$$\begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, \{u_i\}_i, v, \mathsf{aux}, \\ \{v(\mathbf{x}_i), H'_{i-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i, \{\mathbf{r}_i \cdot \mathbf{x}_i\}_{i<k}, b_k \end{pmatrix}$$

We recall that our adversary aborts if $v(\mathbf{x}_i)$ are not distinct. In the case where they are distinct, given one of the samples in the preceding equations, $\mathcal{A}'''$ samples u_j such that $u_j(v(\mathbf{x}_i)) = \mathbf{r}_i$ and that u_j is sampled uniformly according to the d-wise independent sampling procedure. Then $\mathcal{A}'''$ simulates the samples expected by $\mathcal{A}''$. The result is $\mathcal{A}'''$ distinguishes the two cases with probability ϵ'/d.

Now fix $\mathbf{f}, \mathbf{h}, u_1, \ldots, u_{j-1}, v$, which fixes H'_{i-1}. Let $y_i^{(j)} = H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)$. Notice that since $\mathbf{f}, \mathbf{h}, u_1, \ldots, u_{j-1}, v$ are fixed and H'_{j-1} has image size at most r, there are at most r^d possible values for the vector $(y_1^{(j)}, \ldots, y_d^{(j)})$, and recall that r is a polynomial. If d is constant, then r^d is still polynomial. Moreover, there are at most r^d values for the vector $(v(\mathbf{x}_1), \ldots, v(\mathbf{x}_d))$.

Now, we use Lemma 1. Since $\mathbf{x}_k, \mathsf{aux}$ is computationally unpredictable and since there are only a polynomial number of images of v and H'_{j-1}, we have

$$(\mathbf{x}_k, (\ j, k, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_{j-1}, v, \mathsf{aux}, \{v(\mathbf{x}_i), H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i\ ))$$

is computationally unpredictable as well. Even more, it must be that

$$(\mathbf{x}_k, \mathsf{aux}_k = \begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, \{u_i\}_i, v, \mathsf{aux}, \\ \{v(\mathbf{x}_i), H'_{j-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i, \{\mathbf{r}_i, \mathbf{r}_i \cdot \mathbf{x}_i\}_{i<k} \end{pmatrix})$$

is computationally unpredictable, since there are only 2^d possible values to guess for $\mathbf{r}_i \cdot \mathbf{x}_i$.

Therefore, by Goldreich-Levin, we have that $(\mathsf{aux}_k, \mathbf{r}_k, \mathbf{r}_k \cdot \mathbf{x}_k)$ is computationally indistinguishable from $(\mathsf{aux}_k, \mathbf{r}_k, b_k)$ for random b_k. Putting this together in a simple hybrid argument, we have that the following are indistinguishable:

$$\begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_{j-1}, v, \mathsf{aux}, \\ \{v(\mathbf{x}_i), H'_{i-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i, \{\mathbf{r}_i \cdot \mathbf{x}_i\}_{i<k}, \mathbf{r}_k \cdot \mathbf{x}_k \end{pmatrix} \text{ and}$$
$$\begin{pmatrix} j, k, \mathbf{f}, \mathbf{h}, u_1, \ldots, u_{j-1}, v, \mathsf{aux}, \\ \{v(\mathbf{x}_i), H'_{i-1}(u_{[j-1]}(v(\mathbf{x}_i)) \cdot \mathbf{x}_i)\}_i, \{\mathbf{r}_i \cdot \mathbf{x}_i\}_{i<k}, b_k \end{pmatrix}$$

But these are exactly the distributions distinguished by $\mathcal{A}'''$. Therefore, we must have ϵ'/d, and hence ϵ', is negligible. But since $\epsilon' = \epsilon/\ell - 4\delta$ and $\delta \leq \epsilon/5\ell$, we have have that ϵ' is lower bounded by $\delta/5/\ell$ infinitely often, a contradiction. This completes the proof. □

4 Trapdoor ELFs

Here, we define and construct ELFs with a trapdoor, combining the features of LDTFs and ELFs.

Definition 6. *An Trapdoor ELF consists of two algorithms* $\mathsf{TELF.GenInj}$ *and* $\mathsf{TELF.GenLossy}$*, as well as a function* $N = N(M)$ *such that* $\log N$ *is polynomial in* $\log M$*.* $\mathsf{TELF.GenInj}$ *takes as input an integer* M*, and outputs the description of two functions* $f : [M] \to [N]$ *and* $f^{-1} : [N] \to [M] \cup \{\perp\}$ *such that:*

- f, f^{-1} *are computable in time polynomial in the bit-length of their input, namely* $\log M$*.*
- *With overwhelming probability (in* $\log M$*),* $f^{-1}(f(x)) = x$ *for all* $x \in [M]$*. In particular* f *is injective*

$\mathsf{TELF.GenLossy}$ *on the other hand takes as input integers* M *and* $r \in [M]$*. It outputs the description of a function* $f : [M] \to [N]$ *such that:*

- *For all* $r \in [M]$*,* $|f([M])| \leq r$ *with overwhelming probability. That is, the function* f *has image size at most* r*.*
- *For any polynomial* p *and inverse polynomial function* δ *(in* $\log M$*), there is a polynomial* q *such that: for any adversary* $\mathcal{A}$ *running in time at most* p*, and any* $r \in [q(\log M), M]$*, we have that*

$$|\Pr[\mathcal{A}(f) = 1 : (f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)] - \Pr[\mathcal{A}(f) = 1 : f \leftarrow \mathsf{TELF.GenLossy}(M, r)]| < \delta$$

In other words, no polynomial-time adversary $\mathcal{A}$ *can distinguish an injective* f *from an* f *with polynomial image size, in the case that* $\mathcal{A}$ *does not get the trapdoor for* f*.*

We also consider *all-but-some* Trapdoor ELFs, which contain many branches, some of which are lossy:

Definition 7. *An All-but-one Trapdoor ELF consists of algorithms* $\mathsf{TELF.GenInj}$ *and* $\mathsf{TELF.GenLossy}$*, as well as a function* $B = B(M), N = N(M)$ *such that* $\log B, \log N$ *are polynomial in* $\log M$*.* $\mathsf{TELF.GenInj}$ *takes as input an integer* M*, and outputs the description of two functions* $f : [B] \times [M] \to [N]$ *and* $f^{-1} : [B] \times [N] \to [M] \cup \{\perp\}$ *such that:*

- f, f^{-1} *are computable in time polynomial in the bit-length of their input, namely* $\log M$*.*
- *With overwhelming probability (in* $\log M$*), for all branches* $b \in [B]$*, we have that* $f^{-1}(b, f(b, x)) = x$ *for all* $x \in [M]$*. In particular* $f(b, \cdot)$ *is injective*

$\mathsf{TELF.GenLossy}$ *on the other hand takes as input integers* M *and* $r \in [M]$*, and a branch* $b^* \in [B]$*. It outputs the description of functions* $f : [B] \times [M] \to [N]$ *and* $f^{-1} : [B] \times [N] \to [M] \cup \{\perp\}$ *such that:*

- *For all $r \in [M]$, $|f(b, [M])| \leq r$ with overwhelming probability. That is, the function $f(b^*, \cdot)$ has image size at most r.*
- *With overwhelming probability (in $\log M$), for all branches $b \in [B] \setminus \{b^*\}$, $f^{-1}(b, f(b, x)) = x$ for all $x \in [M]$.*
- *For any polynomial p and inverse polynomial function δ (in $\log M$), there is a polynomial q such that: for any adversary $\mathcal{A}$ running in time at most p and playing the following game, it's advantage is at most δ:*
  - *First, A chooses a branch b^*, which is sends to the challenger.*
  - *The challenger then either runs $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)$ or runs $(f, f^{-1} \leftarrow \mathsf{TELF.GenLossy}(M, b^*, r)$, and sends f to A*
  - *A can make queries to f^{-1} on all branches other than b^*.*
  - *A outputs a guess b for which f it was given*

  A's advantage is defined to be the difference

$$|\Pr[\mathcal{A}(f) = 1 : (f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)] - \Pr[\mathcal{A}(f) = 1 : (f, f^{-1}) \leftarrow \mathsf{TELF.GenLossy}(M, b^*, r)|$$

  In other words, no polynomial-time adversary $\mathcal{A}$ can distinguish an injective f from an f where branch b^* has polynomial image size, in the case that $\mathcal{A}$ does not get the trapdoor for f.

An all-but-some Trapdoor ELF generalizes the above to allow the lossy mode to contain multiple lossy branches. We omit the details of the definition.

4.1 Constructing Trapdoor ELFs

Here, we construct Trapdoor ELFs from exponentially-hard DDH, which is plausible on certain elliptic curve groups. Our construction will follow mostly Zhandry's [22] construction of ELFs, with some modifications to obtain a trapdoor.

Zhandry's scheme works as follows: first, he considers a *bounded adversary* ELF, which is secure against only adversaries of an a priori bounded running time. This scheme more or less follows from lossy trapdoor functions in the literature, just pushed into extreme parameter regimes. Then, he iterates the scheme many times, for many different bounds on the adversaries running time. ELF security follows by invoking security for the bounded adversary ELF that is just large enough to fool the given adversary.

We will adopt the same approach. In particular, we will construct a bounded adversary Trapdoor ELF following the LTDFs from the literature. We will trivially inherit the trapdoors from these schemes. Then, we will iterate the construction. Zhandry's construction, in order to remain efficient, must compress the image every after every iteration. This unfortunately means Zhandry's construction does not have a functioning trapdoor. We therefore devise a way to avoid compressing the input, allowing the trapdoor to remain intact.

Bounded Adversary Trapdoor ELFs Here, we define a bounded adversary Trapdoor ELF, which is a Trapdoor ELF where security is guaranteed only against a prior bounded adversaries. The definition follows almost immediately from adapting Zhandry's bounded adversary ELF definition by adding a trapdoor.

Informally, in an ordinary Trapdoor ELF, r can be chosen based on the adversary to be just high enough to fool it. In contrast, in a bounded adversary Orf, r must be chosen independent of the adversary, and then security only applies to adversaries with running time sufficiently smaller than r. Moreover, the adversary gets to learn r.

Definition 8. *An* bounded adversary *Trapdoor ELF consists of two algorithms* $\mathsf{TELF.GenInj}'$ *and* $\mathsf{TELF.GenLossy}'$*, and a function* $N = N(M, r)$*.* $\mathsf{TELF.GenInj}'$ *takes as input an integer* M *and integer* $r \in [M]$ *and outputs the description of two functions* $f : [M] \to [N]$ *and* $f^{-1} : [N] \to [M] \cup \{\perp\}$ *such that:*

- f, f^{-1} *are computable in time polynomial in the bit-length of their input, namely* $\log M$*.*
- *With overwhelming probability (in* $\log M$*),* $f^{-1}(f(x)) = x$ *for all* $x \in [M]$*. In particular* f *is injective*

$\mathsf{TELF.GenLossy}'$ *also takes as input integers* M *and* $r \in [M]$*. It outputs the description of a function* $f : [M] \to [N]$ *such that:*

- *For all* $r \in [M]$*,* $|f([M])| \leq r$ *with overwhelming probability. That is, the function* f *has image size at most* r*.*
- *For any polynomial* p *and inverse polynomial function* δ *(in* $\log M$*), there is a polynomial* q *such that: for any adversary* $\mathcal{A}$ *running in time at most* p*, and any* $r \in [q(\log M), M]$*, we have that*

$$|\Pr[\mathcal{A}(r, f) = 1 : (f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}'(M)] - \Pr[\mathcal{A}(r, f) = 1 : f \leftarrow \mathsf{TELF.GenLossy}'(M, r)| < \delta$$

In other words, no polynomial-time adversary $\mathcal{A}$ *can distinguish an injective* f *from an* f *with polynomial image size, in the case that* $\mathcal{A}$ *does not get the trapdoor for* f*. Unlike an ordinary Trapdoor ELF, this holds* even if the adversary knows r*.*

Constructing Bounded Adversary Trapdoor ELFs Our construction of bounded adversary Trapdoor ELFs, like Zhandry's ELFs, is based on the DDH-based lossy *trapdoor* functions of Peikert and Waters [18] and Freeman et al. [13]. In fact, since Zhandry did not need the trapdoor of prior constructions, the construction for ELFs was very slightly simplified. In contrast, our construction almost verbatim matches the construction Freeman et al., except that the group size is set to be much smaller, in particular polynomial. In order to maintain security in this regime, we must rely on the exponential hardness of the group.

Cryptographic Groups. The following definitions and notation are almost verbatim from Zhandry [22].

Definition 9. *A* cryptographic group *consists of an algorithm* $\mathsf{Group.Gen}$ *which takes in a security parameter* λ*, and produces a (description of a) cyclic group* $\mathbb{G}$ *of prime order* $p \in [2^\lambda, 2 \times 2^\lambda)$*, and a generator* g *for* $\mathbb{G}$ *such that:*

- *The group operation* $\times : \mathbb{G}^2 \to \mathbb{G}$ *is polynomial-time computable in* λ*.*
- *Exponentiation by elements in* $\mathbb{Z}_p$ *is polynomial-time computable in* λ*.*
- *The representation of a group element* h *has size polynomial in* λ*.*

For some notation: given a matrix $\mathbf{A} \in \mathbb{Z}_p^{m \times n}$, we write $g^{\mathbf{A}} \in \mathbb{G}^{m \times n}$ to be the $m \times n$ matrix of group elements $g^{A_{i,j}}$. Analogously define $g^{\mathbf{w}}$ for a vector $\mathbf{w} \in \mathbb{Z}_p^n$. Given a matrix $\hat{\mathbf{A}} \in \mathbb{G}^{m \times n}$ of group elements and a vector $\mathbf{v} \in \mathbb{Z}_p^n$, write $\hat{\mathbf{A}} \cdot \mathbf{v}$ to mean $\hat{\mathbf{w}} \in \mathbb{G}^m$ where $\hat{w}_i = \prod_{j=1}^n \hat{A}_{i,j}^{v_j}$. Using this notation, $(g^{\mathbf{A}}) \cdot \mathbf{v} = g^{\mathbf{A} \cdot \mathbf{v}}$. Therefore, the map $g^{\mathbf{A}}, \mathbf{v} \mapsto g^{\mathbf{A} \cdot \mathbf{v}}$ is efficiently computable.

Definition 10. *The* exponential decisional k-linear assumption *(*k*-eLin) on a cryptographic group specified by* $\mathsf{Group.Gen}$ *holds if there is a polynomial* $q(\cdot\cdot)$ *such that the following is true. For any time bound* t *and probability* ϵ*, let* $\lambda = \log q(t, 1/\epsilon)$*. Then for any adversary* $\mathcal{A}$ *running in time at most* t*, the following two distributions are indistinguishable, except with advantage at most* ϵ*:*

$$(\mathbb{G}, g, g^{a_1}, \ldots, g^{a_k}, g^c, g^{a_1 b_1}, \cdots g^{a_k b_k}) : \begin{array}{l} (\mathbb{G}, g, p) \leftarrow \mathsf{Group.Gen}(\lambda) \\ a_i, b_i, c \leftarrow \mathbb{Z}_p \end{array} \text{, and}$$

$$(\mathbb{G}, g, g^{a_1}, \ldots, g^{a_k}, g^{\sum_{i=1}^k b_i}, g^{a_1 b_1}, \cdots g^{a_k b_k}) : \begin{array}{l} (\mathbb{G}, g, p) \leftarrow \mathsf{Group.Gen}(\lambda) \\ a_i, b_i \leftarrow \mathbb{Z}_p \end{array}$$

$k = 1$ corresponds to the eDDH assumption above.

As a special case, $k = 1$ corresponds to the exponential DDH assumption. A plausible candidate for a cryptographic group supporting the eDDH assumption or k-linear assumption are groups based on elliptic curves. Despite over a decade or research, the best attacks on many elliptic curves are generic attacks which require exponential time. Therefore, the eDDH assumption on these groups appears to be a very reasonable assumption.

Construction. Our construction is as follows, and will be parameterized by k. $\mathsf{TELF.GenInj}'_k(M, r)$ does the following.

- Let λ be the largest integer such that $(2 \times 2^\lambda)^k < r$. Run $(\mathbb{G}, g, p) \leftarrow \mathsf{Group.Gen}(\lambda)$.
- Let m be the smallest integer such that $2^m \geq M$. Let R be an efficiently invertible function from $[M]$ into $\{0,1\}^m$.
- Let $n \geq m$ (e.g. $m = 2n$) be chosen such that a random matrix sampled from $\mathbb{Z}_p^{n \times m}$ has rank m with overwhelming probability. Note that a random *square* matrix will be singular with probability $1/p$, and in our case, p is polynomial. Hence we require m somewhat larger than n.

- Choose a random matrix $n \times m$ matrix $\mathbf{A}$ of elements in $\mathbb{Z}_p^{n \times m}$. Set $\hat{\mathbf{A}} = g^{\mathbf{A}}$.
- Output functions f, f^{-1}. f is defined as $f(x) = \hat{\mathbf{A}} \cdot (R(x))$. The description of f will consist of $(\mathbb{G}, p, \hat{\mathbf{A}}, R, m, n)$.
  f^{-1} is defined as follows. Let $\mathbf{B} \in \mathbb{Z}_p^{m \times n}$ such that $\mathbf{B} \cdot \mathbf{A}$ is the identity. Given a vector $\mathbf{v} \in \mathbb{G}^n$, compute $\mathbf{w} = \mathbf{B} \cdot \mathbf{v}$. Then, try to compute the discrete log of each component by testing if the component is g^0 or g^1. If any of the discrete log computations fail, then output $\perp$. Otherwise, let $\mathbf{y}$ be the vector of exponents obtained. Invert R on y to obtain x. If inversion fails, output $\perp$. Otherwise, output x. The description of f^{-1} will consist of $(\mathbb{G}, p, \mathbf{B}, R, m, n)$.
  $\mathsf{TELF.GenLossy}'_k(M, r)$ is identical to $\mathsf{TELF.GenInj}'_k(M, r)$, except the matrix $\mathbf{A}$ is chosen to be random of rank k, rather than full rank. In this case, there is no $\mathbf{B}$ and hence no function f^{-1}.

Theorem 3. *If* Group.Gen *is a group where the k-eLin assumption holds for some constant k, then* $(\mathsf{TELF.GenInj}'_k, \mathsf{TELF.GenLossy}'_k)$ *is a bounded adversary Trapdoor ELF.*

Proof. For correctness, notice that $\mathbf{w}$ computed by f^{-1} is equal to $\mathbf{B} \cdot \mathbf{v} = \mathbf{B} \cdot \hat{\mathbf{A}} \cdot R(x) = g^{\mathbf{B} \cdot \mathbf{A} \cdot R(x)} = g^{R(x)}$. Therefore, when f^{-1} is given a valid output of f, it will recover $g^{R(x)}$, and the discrete log computations will yield $R(x)$ and the final inversion of R will yield x, as desired.

Security follows from an almost identical argument to the security of bounded adversary ELFs in Zhandry [22], and we only sketch the details here. All that needs to be shown is that $g^{\mathbf{A}}$ for a random matrix is indistinguishable from $g^{\mathbf{A}}$ for a random rank-k matrix. This follows by standard hybrid arguments (e.g. [20]) and the assumed k-linear assumption. □

Constructing Ordinary Trapdoor ELFs We now turn to using bounded adversary Trapdoor ELFs to construct ordinary Trapdoor ELFs. Here, we depart slightly from Zhandry [22]. Zhandry's idea is to iterate many bounded adversary functions as r ranges over the powers of 2. The injective mode just sets all the bounded adversary functions to be injective. For the lossy mode, a single function is set to be lossy, namely the function that is big enough to fool the adversary in question.

One issue that immediately becomes apparent in the above approach is that the bounded adversary functions are expanding. As such, the overall domain will grow exponentially with the number of iterations, leading to an inefficient scheme. Zhandry gets around this by applying a pairwise independent function between each bounded adversary function to compress the output and keep it polynomial in size. Unfortunately, this compression destroys any trapdoor in the bounded adversary function.

Instead, our approach is to *not* compress the outputs, but be very careful about which r we choose for our bounded adversary Trapdoor ELFs. In particular, notice that our bounded adversary Trapdoor ELFs expand the input by a factor of $C_k \times \log r$, for some constant C that depends on k. Therefore, if our construction

uses a sequence $r_1, \ldots, r_t$ of r's, the overall expansion is $C_k^t \prod \log r_i$. We need this expansion factor to be polynomial in size.

Notice that the powers of 2, namely $r_i = 2^i$, used by Zhandry do not work, as the overall expansion will be $C_k^t t!$. We need $r_t = 2^t$ to be larger than any polynomial in our security parameter $\log M$ (so that we can set r based on any adversary), meaning the overall expansion factor will be at least $C_k^{\log\log M}(\log\log M)!$. Notice that $(\log\log M)!$ is super-polynomial in $\log M$, leading to an inefficient scheme.

Instead, we choose $r_i = 2^{2^i}$, and let i go from 1 to $t = \sqrt{\log\log M}$. We see that the overall expansion factor is:

$$\begin{aligned} C_k^t \prod_{i=1}^{t} \log r_i &= C_k^{\sqrt{\log\log M}} \prod_{i=1}^{t} 2^i \\ &\leq C_k^{\log\log M} \prod_{i=1}^{t} 2^i = (\log M)^{\log C_k} \prod_{i=1}^{t} 2^i \\ &= (\log M)^{\log C_k} 2^{\sum_{i=1}^{t} i} \leq = (\log M)^{\log C_k} 2^{t^2} = (\log M)^{1+\log C_k} \end{aligned}$$

We also note that $r_t = 2^{2^{\sqrt{\log\log M}}}$ is larger than any polynomial in $\log M$. This means that for any polynomial p, we can always choose i so that r_i will be at most approximately p^2. This is exactly what we need to argue security.

In more detail, our construction does the following. Assume for the bounded adversary Trapdoor ELF that $N = N(M, r)$ satisfies $\log N \leq C(\log M)(\log r)$ for some universal constant C, as in our bounded adversary construction. Then $\mathsf{TELF.GenInj}(M)$ does the following:

- Let t be the smallest integer such that $2^{2^{t^2}} \geq M$.
- Let $M_1 = M$.
- For $i = 1, \ldots, t$, Run $(f_i, f_i^{-1}) \leftarrow \mathsf{TELF.GenInj}'(M_i, r_i)$ for $r_i = 2^{2^i}$. Let N_i be the output space of f_i, and set $M_{i+1} = N_i$.
- Let $f : [M] \to [N_t]$ be $f_t \circ f_{t-1} \circ \cdots \circ f_1$. Let f^{-1} attempt to compute $f_1^{-1} \circ \ldots f_t^{-1}$, and output $\perp$ if any of the inversions fail.
- Output (f, f^{-1}).

$\mathsf{TELF.GenLossy}(M, r)$ is the same as $\mathsf{TELF.GenInj}$, except that it lets i^* be the largest integer such that $r_{i^*} \leq r$ and $i^* \leq t$. It then computes $f_{i^*} \leftarrow \mathsf{TELF.GenLossy}'(M_{i^*}, r_{i^*})$ instead of using $\mathsf{TELF.GenInj}'$. It lets f be defined as above, and outputs f (but no f^*).

Theorem 4. *If* $(\mathsf{TELF.GenInj}', \mathsf{TELF.GenLossy}')$ *is a bounded-adversary Trapdoor ELF satisfying* $\log N \leq C(\log M)(\log r)$ *for some constant* C*, then we have that* $(\mathsf{TELF.GenInj}, \mathsf{TELF.GenLossy})$ *is an ordinary Trapdoor ELF.*

Proof. The image size in the lossy mode is guaranteed by how we chose i^*. Namely, the image size on input r is at most r_{i^*} which is at most r.

It remains to prove security. Let p be a polynomial and σ be an inverse polynomial in $\log M$. Let p' be p plus the running time of $\mathsf{TELF.GenInj}$. Let q be the polynomial guaranteed by $(\mathsf{TELF.GenInj}', \mathsf{TELF.GenLossy}')$ for p' and σ.

Notice that q will be a polynomial in the $\log M_i$, the domain for the functions $(\mathsf{TELF.GenInj}', \mathsf{TELF.GenLossy}')$, and not in $\log M$. Nonetheless, we can redefine q to be a polynomial in $\log M$ since $\log M_i$ is polynomial in $\log M$.

Consider any adversary A for $(\mathsf{TELF.GenInj}, \mathsf{TELF.GenLossy})$ running in time at most p. Let $r = r(M)$ be a computable function of M such that $r \in (q(\log M), M]$. Our goal is to show that A distinguishes f from $\mathsf{TELF.GenInj}(M)$ from $\mathsf{TELF.GenLossy}(M, r)$ with advantage less than δ.

Toward that goal, let i^* be the largest integer such that $r_{i^*} = 2^{2^{i^*}} \leq r$ and $i^* \leq t$. We construct an adversary A' for $(\mathsf{TELF.GenInj}', \mathsf{TELF.GenLossy}')$ with $r = r_{i^*}$. Let f_{i^*} be the f that A' receives, where f_{i^*} is either $\mathsf{TELF.GenInj}'(M, r_{i^*})$ or $\mathsf{TELF.GenLossy}'(M, r_{i^*})$. Then A' simulates the rest of f for itself, setting $(f_i, f_i^{-1}) \leftarrow \mathsf{TELF.GenInj}'(M_i, r_i)$ for $i \neq i^*$. A' then runs A on the simulated f. Notice that A' runs in time at most p'. Thus by the bounded-adversary security of $(\mathsf{TELF.GenInj}', \mathsf{TELF.GenLossy}')$, A' cannot distinguish injective or lossy mode, except with advantage σ. Moreover, if $f_{i^*} \leftarrow \mathsf{TELF.GenInj}'(M, r_{i^*})$, then this corresponds to $\mathsf{TELF.GenInj}$, and if $f_{i^*} \leftarrow \mathsf{TELF.GenLossy}'(M, r_{i^*})$, then this corresponds to $\mathsf{TELF.GenLossy}(M, r)$. Thus, A' and A have the same distinguishing advantage, and therefore A cannot distinguish the two cases except with probability less than σ. □

4.2 Constructing All-but-some Trapdoor ELFs

We now turn to constructing All-but-some Trapdoor ELFs. It is sufficient to construct a bounded adversary version of All-but-some Trapdoor ELFs, which can then be converted into full All-but-some Trapdoor ELFs using the conversion in the preceding section. Here, we describe how to do this. We focus on the all-but-one case, the all-but-some being a simple generalization.

Construction. Our construction is as follows, and will be parameterized by k. The branch set B will be interpreted as $\{0,1\}^a$ for some polynomial a. $\mathsf{TELF.GenInj}'_k(M, b, r)$ does the following.

- Let λ be the largest integer such that $(2 \times 2^\lambda)^k < r$. Run $(\mathbb{G}, g, p) \leftarrow \mathsf{Group.Gen}(\lambda)$.
- Let m be the smallest integer such that $2^m \geq M$. Let R be an efficiently invertible function from $[M]$ into $\{0,1\}^m$.
- Let $n \geq m$ be chosen such that a random matrix sampled from $\mathbb{Z}_p^{n \times m}$ has rank m with overwhelming probability. Note that a random *square* matrix will be singular with probability $1/p$, and in our case, p is polynomial. Therefore, we need to choose an n somewhat larger than m. It suffices to set $n = 2m$.
- Choose $2a + 1$ random $2n \times m$ matrices $\mathbf{B}, \mathbf{A}_{i,t}$ in $\mathbb{Z}_q^{2n \times m}$, and let $\hat{\mathbf{A}}_{i,t} = g^{\mathbf{A}_{i,t}}, \hat{\mathbf{B}} = g^{\mathbf{B}}$.
  Define $\mathbf{A}_b = \mathbf{B} + \sum_i \mathbf{A}_{i,b_i}$.

– Output functions f, f^{-1}. f is defined as $f(b, x) = \hat{\mathbf{A}}_b \cdot (R(x))$. Note that $\hat{\mathbf{A}}_b$ can be computed from $\hat{\mathbf{A}}_{i,t}, \hat{\mathbf{B}}$. The description of f will consist of $(\mathbb{G}, p, \hat{\mathbf{B}}, \{\hat{\mathbf{A}}_{i,t}\}, R, m, n)$.
$f^{-1}(b, v)$ is defined as follows. Let $\mathbf{A}_b^{-1} \in \mathbb{Z}_p^{m \times 2n}$ such that $\mathbf{A}_b^{-1} \cdot \mathbf{A}_b$ is the identity. Given a vector $\mathbf{v} \in \mathbb{G}^n$, compute $\mathbf{w} = \mathbf{A}_b^{-1} \cdot \mathbf{v}$. Then, try to compute the discrete log of each component by testing if the component is g^0 or g^1. If any of the discrete log computations fail, then output $\perp$. Otherwise, let $\mathbf{y}$ be the vector of exponents obtained. Invert R on y to obtain x. If inversion fails, output $\perp$. Otherwise, output x. The description of f^{-1} will consist of $(\mathbb{G}, p, \mathbf{B}, \{\mathbf{A}_{i,t}\}, R, m, n)$.
$\mathsf{TELF.GenLossy}'_k(M, b, r)$ is identical to $\mathsf{TELF.GenInj}'_k(M, b, r)$, except the matrix $\mathbf{A}_b$ is chosen to be random of rank k, rather than full rank. Then $\mathbf{B}$ is set to $\mathbf{A}_b - \sum_i \mathbf{A}_{i,b_i}$.

Theorem 5. *If* $\mathsf{Group.Gen}$ *is a group where the* k*-eLin assumption holds for some constant* k*, then* $(\mathsf{TELF.GenInj}'_k, \mathsf{TELF.GenLossy}'_k)$ *is a bounded adversary all-but-one Trapdoor ELF.*

Proof. We just need to show, given a branch b^*, how to embed a challenge $g^{\mathbf{C}}$ into the description of f so that:

– If $\mathbf{C}$ is full rank, $\mathbf{B}, \mathbf{A}_{i,t}$ is distributed as in the injective mode, namely uniformly random.
– If $\mathbf{C}$ has rank k, then $\mathbf{B}, \mathbf{A}_{i,t}$ is distributed as in the lossy mode for branch b^*, namely $\mathbf{A}_{b^*}$ is random of rank k.
– We can simulate inversion queries on all other branches.

To do so, we exploit the fact that we have some extra rows to work with. We will assume the challenge $g^{\mathbf{C}}$ is $n \times m$. We will choose a uniformly random matrix $\mathbf{S} \in \mathbb{Z}_p^{2n \times 2n}$. We will set $\mathbf{A}'_{b^*}$ to be the block matrix with $\mathbf{C}$ on top, and $0^{2n \times m}$ on bottom. Then we will set $\mathbf{A}_{b^*} = \mathbf{S} \cdot \mathbf{A}'_{b^*}$.

We will choose $\mathbf{A}'_{i,t}$ as random $2n \times m$ matrices, and then set $\mathbf{A}_{i,t} = \mathbf{S} \cdot \mathbf{A}'_{i,t}$. Finally, we will set $\mathbf{B} = \mathbf{A}_{b^*} - \sum_i \mathbf{A}_{i,b_i^*} = \mathbf{S} \cdot \left(\mathbf{A}'_{b^*} - \sum_i \mathbf{A}'_{i,b_i^*}\right)$.

We can now compute $g^{\mathbf{A}_{i,t}}$ using our knowledge of $\mathbf{A}_{i,t}$, and $g^{\mathbf{B}}$ using our knowledge of $\mathbf{A}_{i,t}$, $\mathbf{S}$, and $g^{\mathbf{C}}$.

It is straightforward to show that if $\mathbf{C}$ is a uniformly random matrix, then so are all the matrices $\mathbf{B}, \mathbf{A}_{i,t}$. Moreover, if $\mathbf{C}$ is random of rank k, is is straightforward that the matrices are random, subject to $\mathbf{A}_{b^*}$ being rank k, as desired.

It remains to prove that we can answer inversion queries. Here, we simply use the fact that we know the bottom $n \times m$ matrices in the clear, meaning we can perform the inversion operation as in standard Trapdoor ELFs. As the last step, we just verify our inversion by evaluating the Trapdoor ELF on the derived pre-image, ensuring that it matches the provided image point. □

We can easily use the above techniques to extend to ℓ lossy branches in several ways. One way is to simply evaluate ℓ different Trapdoor ELFs in sequence; to set the ℓ different branches, simply assign one branch to each of the Trapdoor ELFs.

5 DPKE for Computationally Unpredictable Sources

In this section, we show our basic DPKE construction, a deterministic public key encryption scheme (DPKE) for arbitrary computational sources.

The Construction. The message space for our scheme is $[M]$. We will use a hardcore function $\mathcal{G}$ with domain $[M]$, a PKE scheme $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$, and a trapdoor ELF $(\mathsf{TELF.GenInj}, \mathsf{TELF.GenLossy})$.

- $\mathsf{DPKE.Gen}$ runs $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{PKE.Gen}(\lambda)$, $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)$, and $G \leftarrow \mathcal{G}$. It outputs $\mathsf{sk} = (\mathsf{sk}', f^{-1})$ and $\mathsf{pk} = (\mathsf{pk}', f, G)$.
- $\mathsf{DPKE.Enc}(\mathsf{pk}, m)$ runs $\mathsf{PKE.Enc}(\mathsf{pk}', f(m); G(m))$. That is, it encrypts $f(m)$ under the semantically secure encryption scheme, using random coins $G(m)$
- $\mathsf{DPKE.Dec}(\mathsf{sk}, c)$: run $y \leftarrow \mathsf{PKE.Dec}(\mathsf{sk}', c)$. If $y = \perp$ output $\perp$. Otherwise run $m \leftarrow f^{-1}(y)$ and output m.

Correctness of the scheme is immediate. For security, we have the following theorem:

Theorem 6. *For any constant d, if $\mathcal{G}$ is hardcore for arbitrary computationally unpredictable sources on d inputs,* $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is semantically secure, and* $(\mathsf{TELF.GenInj}, \mathsf{TELF.GenLossy})$ *is a secure Trapdoor ELF, then* $(\mathsf{DPKE.Gen}, \mathsf{DPKE.Enc}, \mathsf{DPKE.Dec})$ *is a secure deterministic public key encryption scheme for arbitrary single computationally unpredictable sources on d inputs. If* $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *has pseudorandom ciphertexts, then so does* $(\mathsf{DPKE.Gen}, \mathsf{DPKE.Enc}, \mathsf{DPKE.Dec})$.

Proof. Consider an arbitrary computationally unpredictable source D, sampling messages $m_1, \ldots, m_d$ and auxiliary information aux. We will prove the pseudorandom ciphertext case, the other case being analogous. We need to prove that $(\mathsf{pk}, \mathsf{DPKE.Enc}(\mathsf{pk}, m_1), \ldots, \mathsf{DPKE.Enc}(\mathsf{pk}, m_d), \mathsf{aux})$ is computationally indistinguishable from $(\mathsf{pk}, C_1, \ldots, C_d, \mathsf{aux})$, where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{DPKE.Gen}(\lambda)$, $(m_1, \ldots, m_d, \mathsf{aux}) \leftarrow D$, and C_i are chosen uniformly random from the ciphertext space.

Suppose toward contradiction that we have an adversary A which distinguishes the two distributions with advantage ϵ. Let p be a polynomial such that $1/p \geq \epsilon$ infinitely often. We prove security through a sequence of hybrids:

- H_0. In this hybrid, the adversary is given $(\mathsf{pk}, c_1, \ldots, c_d, \mathsf{aux})$ where $\mathsf{pk} = (\mathsf{pk}', f, G)$, $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{PKE.Gen}(\lambda)$, $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)$, $G \leftarrow \mathcal{G}$, and $c_i = \mathsf{DPKE.Enc}(\mathsf{pk}, m_i) = \mathsf{PKE.Enc}(\mathsf{pk}', f(m_i); G(m_i))$
- H_1. In this hybrid, we change f to be lossy. That is we choose r so that A cannot distinguish $f \leftarrow \mathsf{TELF.GenLossy}(M, r)$ from f, except with probability $1/3p$. We then replace f with $f \leftarrow \mathsf{TELF.GenLossy}(M, r)$.
- H_2. In this hybrid, we change $c_i = \mathsf{PKE.Enc}(\mathsf{pk}', f(m_i); G(m_i))$ to $c_i = \mathsf{PKE.Enc}(\mathsf{pk}', f(m_i); R_i)$. That is, we replace $G(m_i)$ with R_i. We now claim that A distinguishes H_1 from H_2 with negligible probability.

To prove this, notice that by Lemma 1 and the fact that d is constant, we have that $(m_1, \ldots, m_d,\ (\mathsf{aux}, f, f(m_1), \ldots, f(m_d)\ ))$ is also computationally unpredictable. Then by the hardcore-ness of $\mathcal{G}$, we have that

$$(G(m_1), \ldots, G(m_d),\ (\mathsf{aux}, G, f, f(m_1), \ldots, f(m_d)\ ))$$

is indistinguishable from

$$(R_1, \ldots, R_d,\ (\mathsf{aux}, G, f, f(m_1), \ldots, f(m_d)\ ))$$

Finally by post-processing with $\mathsf{PKE.Enc}$, we have that

$$(\{\mathsf{PKE.Enc}(\mathsf{pk}, f(m_i); G(m_i))\}, \mathsf{aux}, G, f, \{f(m_i)\}, \mathsf{pk})$$

is indistinguishable from

$$(\{\mathsf{PKE.Enc}(\mathsf{pk}, f(m_i); R_i)\}, \mathsf{aux}, G, f, \{f(m_i)\}, \mathsf{pk})$$

The first case is H_1, and the second is H_2, proving their indistinguishability.

- H_3. Now we just change each c_i to be a uniformly random ciphertext C_i. The indistinguishability from H_2 follows from the pseudorandomness of $\mathsf{PKE.Enc}$.
- H_4. Finally, we change f back to the injective mode, generating $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)$ By analagous arguments, A distinguishes H_4 from H_3 with advantage $1/3p$. The result is that the adversary now sees $(\mathsf{pk}, C, \mathsf{aux})$

Putting it all together, A distinguishes H_0 from H_4 with advantage at most $2/3p - \mathsf{negl} \leq 1/p \leq \epsilon$, a contradiction. □

6 Achieving CCA security

In this section, we turn to building CCA-secure DPKE for computationally unpredictable sources.

We will loosely follow Peikert and Waters [18], who build CCA-secure public key encryption from lossy trapdoor functions (LTDFs). The main difficulty is that we want to switch to lossy mode in order to prove the security of the challenge ciphertext, but need to maintain the ability to decrypt all other ciphertexts. Their core idea is to devise a LTDF with many "branches", each ciphertext using a different branch. The challenge ciphertext is set to be encrypted using a lossy, and all others are injective.

We will use this idea, but the technical implementation will be somewhat different, and of course we will use a Trapdoor ELF with branches instead of an LTDF. The details are below.

6.1 Our Construction

Our building blocks will be a pseudorandom generator G, a CCA-secure public key encryption scheme $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$, and an all-but-one Trapdoor ELF $(\mathsf{TELF.GenInj}', \mathsf{TELF.GenLossy}')$.

– $\mathsf{DPKE.Gen}$ runs $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{PKE.Gen}(\lambda)$, $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}'(M)$, and $G_0, G_1 \leftarrow \mathcal{G}$. It outputs $\mathsf{sk} = (\mathsf{sk}', f^{-1})$ and $\mathsf{pk} = (\mathsf{pk}', f, G_0, G_1)$.
– $\mathsf{DPKE.Enc}(\mathsf{pk}, m)$ runs $b \leftarrow G_0(m)$ to select a branch. Then it applied our scheme from Section 5, using the branch b. Namely, it computes $d \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}', f(b, m); G_1(m))$. The output is the ciphertext $c = (b, d)$
– $\mathsf{DPKE.Dec}(\mathsf{sk}, c)$: run $y \leftarrow \mathsf{PKE.Dec}(\mathsf{sk}', c')$. If $y = \perp$ output $\perp$. Otherwise, it runs $m \leftarrow f^{-1}(b, y)$. Finally, it checks that the ciphertext is well-formed by re-encrypting m. Namely, it verifies that $b = G_0(m)$ and $d = \mathsf{PKE.Enc}(\mathsf{pk}, f(b, m); G_1(m))$. If the checks fail, it outputs $\perp$. Otherwise, it outputs m.

The completeness of the scheme is immediate. Next, we prove security

Theorem 7. *For any constant d, if $\mathcal{G}$ is an injective hardcore function for any computationally unpredictable sources on d inputs,* $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is CCA-secure,* $(\mathsf{TELF.GenInj}, \mathsf{TELF.GenLossy})$ *is a secure all-but-d Trapdoor ELF, then* $(\mathsf{DPKE.Gen}, \mathsf{DPKE.Enc}, \mathsf{DPKE.Dec})$ *is a CCA-secure deterministic public key encryption scheme for arbitrary computationally unpredictable sources on d inputs. If* $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *has pseudorandom ciphertexts, then so does* $(\mathsf{DPKE.Gen}, \mathsf{DPKE.Enc}, \mathsf{DPKE.Dec})$.

Proof. For simplicity, we prove the case $d = 1$, the more general case being a straightforward adaptation. Consider an arbitrary computationally unpredictable source D, sampling messages m and auxiliary information aux. We will prove the pseudorandom ciphertext case, the other case being analogous. We need to prove that $(\mathsf{pk}, \mathsf{DPKE.Enc}(\mathsf{pk}, m), \mathsf{aux})$ is computationally indistinguishable from $(\mathsf{pk}, C, \mathsf{aux})$, where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{DPKE.Gen}(\lambda)$, $(m, \mathsf{aux}) \leftarrow D$, and C is chosen uniformly random from the ciphertext space. This must hold even if an adversary can make decryption queries on any ciphertext except the challenge.

Suppose toward contradiction that we have an adversary A which distinguishes the two distributions with advantage ϵ. Let p be a polynomial such that $1/p \geq \epsilon$ infinitely often. We prove security through a sequence of hybrids:

– H_0. Here, we give the adversary $(\mathsf{pk}, c^*, \mathsf{aux})$ where $\mathsf{pk} = (\mathsf{pk}', f, G_0, G_1)$, $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{PKE.Gen}(\lambda)$, and $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)$, and $G_0, G_1 \leftarrow \mathcal{G}$. Also, we set $c^* = \mathsf{DPKE.Enc}(\mathsf{pk}, m) = (b^*, d^*)$ where $b^* = G_0(m)$ and $d^* = \mathsf{PKE.Enc}(\mathsf{pk}', f(b^*, m); G_1(m))$.
– H_1. In this hybrid, we change f to be lossy on the branch b^*. That is, $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M, b^*, r)$, where r is chosen so that A cannot distinguish this change except with advantage $1/3p$.
We need to make sure that we can still answer CCA queries. For this, we just need that G_0 is injective, so that any other valid ciphertext will correspond to a different branch.
– H_2. In this hybrid, we replace $G_1(m)$ with random. We now claim that this change is indistinguishable to the adversary.
Toward that end, first observe that since G_0 is hardcore, we have that $(G_0, G_0(m), \mathsf{aux})$ is indistinguishable from (G_0, S, aux) for a uniformly random

S. This means that $(m,\ (\mathsf{aux}, G_0, G_0(m))\ )$ is computationally unpredictable. But then by Lemma 1, we also have that $(m, (\mathsf{aux}, G_0, b^*, f, f^{-1}, f(b^*, m)))$ is computationally unpredictable, where $(f, f^{-1}) \leftarrow \mathsf{TELF.GenLossy}(M, b^*, r)$ for $b^* = G_0(m)$. Finally, by the hardcore property of G_1, we have that the distribution $(G_1, G_1(m), \mathsf{aux}, G_0, b^*, f, f^{-1}, f(b^*, m))$ is indistinguishable from $(G_1, R, \mathsf{aux}, G_0, b^*, f, f^{-1}, f(b^*, m))$ for a random R.
Now notice that an adversary given $(G_1, R, \mathsf{aux}, G_0, b^*, f, f^{-1}, f(b^*, m))$ for $R = G_1(m)$ (resp. random) can easily simulate the view of A in H_1 (resp. H_2) by using f^{-1} to answer decryption queries. Therefore, if A distinguishes the two hybrids, we can easily create a distinguisher for these two distribution, arriving at a contradiction.

- H_3. Now we just change c to be a uniformly random ciphertext C. The indistinguishability from H_2 follows from the CCA-secure pseudorandomness of $\mathsf{PKE.Enc}$.
  Now notice that the d^* portion of the adversary's view is completely independent of m.
- H_4. Now we invoke the hardcore-ness of G_0 one more time to replace $G_0(m)$ with a random b^*.
- H_5. Finally, we change f back to the injective mode, generating $(f, f^{-1}) \leftarrow \mathsf{TELF.GenInj}(M)$ By analagous arguments, A distinguishes H_5 from H_4 with advantage $1/3p$. The result is that the adversary now sees $(\mathsf{pk}, C, \mathsf{aux})$

Putting it all together, A distinguishes H_0 from H_5 with advantage at most $2/3p - \mathsf{negl} \leq 1/p \leq \epsilon$, a contradiction. □

References

1. J. Alwen, Y. Dodis, and D. Wichs. Survey: Leakage resilience and the bounded retrieval model. In K. Kurosawa, editor, *ICITS 09*, volume 5973 of *LNCS*, pages 1–18. Springer, Heidelberg, Dec. 2010.
2. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Heidelberg, Aug. 2001.
3. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, Heidelberg, Aug. 2007.
4. M. Bellare, D. Cash, and R. Miller. Cryptography secure against related-key attacks and tampering. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 486–503. Springer, Heidelberg, Dec. 2011.
5. M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 360–378. Springer, Heidelberg, Aug. 2008.
6. M. Bellare and V. T. Hoang. Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 627–656. Springer, Heidelberg, Apr. 2015.

7. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, Nov. 1993.
8. A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 335–359. Springer, Heidelberg, Aug. 2008.
9. Z. Brakerski and G. Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 543–560. Springer, Heidelberg, Aug. 2011.
10. C. Brzuska, P. Farshim, and A. Mittelbach. Indistinguishability obfuscation and UCEs: The case of computationally unpredictable sources. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 188–205. Springer, Heidelberg, Aug. 2014.
11. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.
12. K.-M. Chung, H. Lin, M. Mahmoody, and R. Pass. On the power of nonuniformity in proofs of security. In R. D. Kleinberg, editor, *ITCS 2013*, pages 389–400. ACM, Jan. 2013.
13. D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. In P. Q. Nguyen and D. Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 279–295. Springer, Heidelberg, May 2010.
14. B. Fuller, A. O'Neill, and L. Reyzin. A unified approach to deterministic encryption: New constructions and a connection to computational entropy. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 582–599. Springer, Heidelberg, Mar. 2012.
15. O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.
16. V. Goyal, A. O'Neill, and V. Rao. Correlated-input secure hash functions. In Y. Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 182–200. Springer, Heidelberg, Mar. 2011.
17. T. Matsuda and G. Hanaoka. Chosen ciphertext security via UCE. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 56–76. Springer, Heidelberg, Mar. 2014.
18. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *40th ACM STOC*, pages 187–196. ACM Press, May 2008.
19. A. Raghunathan, G. Segev, and S. P. Vadhan. Deterministic public-key encryption for adaptively chosen plaintext distributions. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 93–110. Springer, Heidelberg, May 2013.
20. J. L. Villar. Optimal reductions of some decisional problems to the rank problem. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 80–97. Springer, Heidelberg, Dec. 2012.
21. D. Wichs. Barriers in cryptography with weak, correlated and leaky sources. In R. D. Kleinberg, editor, *ITCS 2013*, pages 111–126. ACM, Jan. 2013.
22. M. Zhandry. The magic of ELFs. In M. Robshaw and J. Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 479–508. Springer, Heidelberg, Aug. 2016.