Encryption Security Against Key-Dependent-Message Attacks: Applications, Realizations and Separations

by

Mohammad Hajiabadi
B.Sc., Sharif University of Technology, 2009
M.Sc., University of Victoria, 2011

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Encryption Security Against Key-Dependent-Message Attacks: Applications, Realizations and Separations

by

Mohammad Hajiabadi
B.Sc., Sharif University of Technology, 2009
M.Sc., University of Victoria, 2011

Supervisory Committee

Dr. Bruce M. Kapron, Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Departmental Member
(Department of Computer Science)

Dr. Audrey Yap, Outside Member
(Department of Philosophy)

Supervisory Committee

Dr. Bruce M. Kapron, Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Departmental Member
(Department of Computer Science)

Dr. Audrey Yap, Outside Member
(Department of Philosophy)

ABSTRACT

In this thesis we study the notion of circular security for bit-encryption schemes. Informally speaking, a bit-encryption scheme is circular secure if it remains secure even if the key of the system is used to encrypt its own individual bits. This notion (or slight extensions thereof) has foundational applications, most notably in the context of fully-homomorphic encryption and amplification techniques for key-dependent-message security.

We explore the notion of circular security from three different perspectives, stemming from (1) assumptions sufficient to realize this notion, (2) minimal black-box assumptions on which this notion can be based and (c) applications of this notion when combined with other properties. Our main results are as follows:

- We give a construction of circular-secure public-key bit encryption based on any public-key encryption scheme that satisfies two special properties. We show that our constructed scheme besides circular security also offers two forms of key-leakage resilience. Our construction unifies two existing specific constructions of circular-secure schemes in the literature and also gives rise to the first construction based on homomorphic hash proof systems.

- We show that seed-circular-secure public-key bit-encryption schemes cannot be based on semantically-secure public-key encryption schemes in a fully-black-box way. A scheme is seed-circular-secure if it allows for the bits of the seed (used to generate the public/secret keys) to be securely encrypted under the corresponding public key. We then extend this result to rule out a large and non-trivial class of constructions for circular security that we call *key-isolating constructions*.

- We give generic constructions of several fundamental cryptographic primitives based on a public-key bit-encryption scheme that combines circular security with a structural property called reproducibility. The main primitives that we build include families of trapdoor functions with strong security properties (i.e., one-wayness under correlated inputs), adaptive-chosen-ciphertext (CCA2) secure encryption schemes and deterministic encryption schemes.

Contents

List of Tables

ACKNOWLEDGEMENTS

I would like to start by thanking my advisor, Bruce Kapron, for all his encouragement and support throughout my years at UVic. There were more than a few times that his encouragement would put me back on track when I was feeling too disappointed with fruitless attempts. Bruce is also among the most liberal individuals I have met, with keen interests in social and political topics that made our (rare) non-technical discussions fun and informative.

I am grateful to Venkatesh Srinivasan for many conversations, technical or otherwise, we had. He did a great job of putting up with my endless complaints about academic bureaucracies, or such things, and he even seemed to (or pretended to?) agree with me sometimes.

I would also like to thank Valerie King for many interesting courses she taught, and for her encouragement during times I was struggling with my research problems. It is really hard not to get inspired by her genuine enthusiasm for research.

I would like to thank both Venkatesh Srinivasan and Audrey Yap for sitting on my committee and for their helpful suggestions. I am grateful to Rafael Pass for agreeing to be the external examiner of my thesis and also for excellent lectures he (along with a few other speakers) gave during a summer school in Bertinoro two years ago, which inspired my interest in black-box separations.

My thanks also go to Josh Benaloh, Dan Boneh and Mohammad Mahmoody for several useful discussions.

Many thanks to Wendy Beggs, Nancy Chan and Jennifer Knapp whose patience and help greatly simplified the whole administrative process.

To all my friends in Victoria: Thank you! My previous lab-mates: Thanks for making our workplace fun. Our exotic hurdle-tuck-jumps over literally any challenging object in the lab not only resulted in an injury in one of my knees, but also made me understand my "limits"!

Finally, I would to thank my parents and siblings for all their love, encouragement and never-ending support in all stages of my life. This thesis is dedicated to them.

I Have Not Failed. I've Just Found 10,000 Ways That Won't Work.
Thomas Edison

Chapter 1

Introduction

1.1 Prelude: Modern Cryptography and Assumption Classification

A central goal in cryptography is to harness the apparent hardness of various computational problems to design protocols (primitives) that meet various goals of security such as secrecy, authenticity, anonymity, etc. The very first step involved in this process is to rigorously formulate the underlying cryptographic primitive, defining what is considered to be a violation of the security goal and what is assumed about the adversary's power. Since the existence of most modern cryptographic primitives imply $\mathbf{P} \neq \mathbf{NP}$ it is beyond the reach of current theory to unconditionally realize these primitives. Instead, a goal set by cryptographers is to build complex cryptographic primitives from relatively simpler ones that are assumed to exist. The eventual ideal goal is to base any cryptographic primitive on the (respective) minimal assumption.

Assumptions in cryptography can roughly be classified into two categories: (a) *specific* assumption, concerning the conjectured hardness of some algebraic computational problem (e.g., factoring products of two large random primes), and (b) *general* assumptions, concerning the conjectured hardness of some general task of computation (e.g., the existence of functions that are easy to compute but hard to invert on average). Often, a cryptographic goal is first realized using specific assumptions and later on solved based on general assumptions. Examples of such success include constructions of pseudorandom number generators (PRGs) based on one-way permutations [95] and one-way functions [61], subsuming those based on the discrete logarithm [17] and factoring [54] assumptions; construction of non-interactive com-

putational zero knowledge for **NP** based on (enhanced) trapdoor permutations [40], generalizing those based on specific number-theoretic assumptions [16, 15]; constructions of statistically-hiding commitment schemes based on collision-resistant hash functions [34, 76], one-way permutations [74] and one-way functions [56], extending those based on specific assumptions [22, 27]; and many more.

Having constructions of a primitive based on general assumptions is important for several reasons:

1. It provides deeper insights into the security properties required by the primitive. In contrast, algebraic aspects of a specific construction that are irrelevant to the construction's security tend to obscure the core ideas that went into the design.

2. It serves as a basis for realizing the primitive based on a wide range of specific assumptions. This is especially useful since future advances in cryptanalysis may invalidate the conjectured hardness of certain specific assumptions.

3. Typically, stronger assumptions lead to more efficient constructions. For examples, even though one-way permutations based PRGs [95, 50] are not practically efficient, they are extremely more efficient than those based on one-way functions [61]. Having constructions based on a variety of general assumptions allows us to understand better the trade-off between the strength of the underlying assumption and the level of efficiency achieved.

As it turns out, not all cryptographic primitives have the same complexity. While many of them are known to be implied by the minimal assumption of the existence of one-way functions (e.g., private-key encryption [49], digital signatures [87], etc), others seem to require substantially stronger assumptions (e.g., key-agreement protocols [67], collisions-resistant hash functions [91], etc.). Thus, an important direction in cryptography is to understand what assumptions are too weak to imply a desired primitive. Note that formalizing this last question requires a new model since all "reasonable" primitives are widely believed to exist and so the implication is widely-believed to hold in a logical sense. Another way to view this is that ruling out these implications in a logical sense immediately imply $\mathbf{P} \neq \mathbf{NP}$ since if $\mathbf{P} = \mathbf{NP}$ all these implications will be vacuously true. We elaborate more on this below.

1.1.1 Black-Box Reductions

To prove (in a mathematical sense) that primitive P_1 implies primitive P_2 (or P_2 *reduce* to P_1) it suffices to give a construction algorithm C and a security-proof algorithm R in such a way that $C(M_1)$ gives an efficient implementation of P_2 whenever M_1 is an efficient implementation of P_1, and that R reduces any efficient attack A_2 against $C(M_1)$ to one against M_1. Most cryptographic reductions are *black-box* in that C only uses M_1 as an *oracle* (denoted C^{M_1}) without assuming anything beyond the input-output correctness of M_1; moreover, R reduces any attack A_2 against C^{M_1} to one against M_1, while treating both M_1 and A_2 as oracles. Such a pair (C, Red) for a cryptographic implication is referred to as a *fully-black-box reduction* [86].

Impagliazzo and Rudich [67] were the first to formalize a model in which to show that certain cryptographic implications cannot be proved in a fully-black-box way. In particular, they showed a fully-black-box separation between secret-key agreement protocols and one-way functions.[1] The framework of [67] was subsequently used (and extended) to separate many cryptographic primitives from many others in a fully-black-box way, e.g., [67, 91, 46, 48, 21, 55, 92, 88, 69]. Some subsequent works showed separations (between certain primitives) with respect to reductions under which only the security proof is required to be black-box (and the construction could be non-black-box) [80, 45, 81].

Although the vast majority of cryptographic techniques are black-box there are also some known non-black-box techniques. These include techniques based on generic zero-knowledge proofs [52, 40], certain techniques based on fully-homomorphic encryption (FHE) [43] and based on indistinguishability obfuscation [8]. For example, Gentry's bootstrapping technique is non-black-box as the constructed FHE-evaluation function makes non-black-box use of the base decryption function. Also, almost all reductions proved in [90] are non-black-box. Some works started to explore the limitations of some of these non-black-box techniques, by showing how to capture them inside a fully-black-box model [25, 5, 6]. For example, the results of Asharov and Segev [5, 6] only rule out fully-black-box reductions to their base primitives, but their base primitives are designed in such a way that the derived results also rule out common non-black-box techniques associated with the use of indistinguishability obfuscators (i.e., those based on the so called punctured programming approach [90]).

[1] We remark that the results of [67] also rule out more general reductions, including relativizing reductions and so-called semi-black-box reductions (following the terminology of [86]). For the latter, they relied on the assumption $\mathbf{P} \neq \mathbf{NP}$, which later became unconditional by Reingold *et al.* [86].

From a practical point of view, non-black-box techniques, due to their essential reliance on the code of their base primitives, lead to much slower constructions, compared to those based on black-box techniques. Moreover, since all known non-black-box techniques are limited to certain domains, if a black-box construction for a desired implication is proven to be impossible, this is generally regarded as strong evidence that proving the implication is currently too difficult (if not impossible) to do.

1.2 Historical Background and Motivations

In this thesis we study the notion of circular security for bit-encryption schemes from different perspectives. To motivate and place our results in context, we first start by providing some historical background.

Secure encryption is one of the most fundamental cryptographic primitives. This primitive, starting with the seminal paper of Goldwasser and Micali [53], has been defined with respect to successively stronger models of security. However, standard notions of encryption security (e.g., semantic (CPA) security and the two forms of chosen-ciphertext (CCA) security [53, 77, 84, 38]) fall short in certain applications, in particular, where the adversary may obtain some side information about the internal secret parameters (e.g., the secret key) of the scheme. This leakage of side information may occur due to some unforeseen attacks on the scheme (*side-channel attacks*), or more fundamentally, when encryption is used as a primitive in a complex protocol which may by design expose inside information. These observations have led to the definition and realization of stronger notions of encryption security, such as security against different forms of key-leakage [71, 39, 1, 75, 36, 35, 2, 60, 23], and key-dependent message (KDM) security [14, 65, 20, 9, 4, 23, 70, 3, 24].

We briefly and informally review some of these notions studied in this thesis, including motivations behind them. For all definitions below (unless otherwise stated) we assume we are encrypting the secret key (or functions thereof) bit-by-bit, i.e., the secret key is a bitstring and the scheme is either bit encryption, or there is a mapping from bits to two fixed plaintext messages.

KDM security. KDM security is defined with respect to a function family F: informally speaking, an encryption scheme (G, E, Dec) is F-KDM secure if the scheme remains semantically secure against any adversary that can ask to obtain an encryption of $f(sk)$, for any $f \in F$ of the adversary's choice. A basic form of KDM security

is circular security, allowing the adversary to obtain encryptions of any bit of the secret key. Another basic notion is projection security, which also allows the adversary to obtain encryptions of negations of secret key bits. (We remark that all these notions can be defined with respect to many pairs of public/secret keys but we ignore this technical point in the introduction.)

KDM security was originally defined by Black *et al.* [14], who built a *fully-KDM*-secure scheme (i.e., KDM-security with respect to all functions) in the random oracle model. In [20] Boneh gave the first construction in the standard model, based on the DDH assumption, of a public-key scheme that they proved KDM secure with respect to *affine functions*. This positive result led to a series of subsequent works, focusing on building affine-KDM security under alternate specific assumptions (i.e., LPN/LWE [4], and QR/DCR [23]), and on developing *KDM-amplification* methods for transforming schemes with basic forms of KDM security into schemes with more sophisticated forms of KDM security [9, 24, 3]. These amplification methods in turn employ techniques such as garbled circuits [9], *randomized encoding of functions* [3] and *entropic-KDM* security [24] to enable KDM transformations. Most relevant to our work are the results of Applebaum [3], showing that, informally speaking, projection security is sufficient to obtain KDM security with respect to any fixed circuit family whose size is poly-bounded. Thus, a fundamental question regarding KDM security is to study general assumptions sufficient for realizing projection security, which is one of the main goals in this thesis.

Another application of the notion of circular security is in the context of *homomorphic bootstrapping* [43], used to transform *bootstrappable somewhat homomorphic* schemes into FHE schemes. Despite the foundational nature of the notion of circular security, our understanding of what it takes to (generically) obtain circular secure encryption is still lacking. The only previous results that address this problem are the work of Haitner and Holenstein [55], which rules out fully-black-box constructions of KDM-secure encryption with respect to quite large function families from trapdoor permutations, and the work of Rothblum [89], which shows no fully-black-box reduction can prove the implication whether "semantic security of a bit-encryption scheme implies circular security of the same scheme." We stress that the result of [89] only considers reductions (from circular security to semantic security) within the same scheme, and not talking about constructions.

Key-leakage resilience. Akavia *et al.* [1] introduce the notion of encryption secu-

rity against bounded memory leakage, wherein an adversary (after seeing the public key) may obtain arbitrary information about the secret key, of the form $f(sk)$ for adaptively chosen f, as long as the total number of bits leaked does not exceed an *a priori* fixed quantity, ℓ. (We refer to the fraction $\ell/|sk|$ as the resilience rate.) They showed that Regev's scheme [85] and the identity based encryption scheme of [44], both under the LWE assumption, provide resilience rate $O(1/polylog(|sk|))$. Naor and Segev [75] showed how to obtain encryption schemes resilient to high leakage lengths (but with low resilience rates) from any hash-proof system [33] and how to obtain schemes with $(1 - o(1))$-resilience rates from d-linear assumptions; moreover, they showed that the circular-secure scheme of [20] provides a $(1-o(1))$ resilience rate. Brakerski and Goldwasser [23], under the subgroup indistinguishability assumption, implied in turn by the QR and DCR assumptions, showed how to obtain encryption schemes that are affine-KDM secure, with a $(1 - o(1))$ resilience rate.

Auxiliary-input security. In the auxiliary-input model [36, 35] the adversary is given some side information of the form $h(pk, sk)$, and the goal is to guarantee security as long as recovering sk from $h(pk, sk)$ is sufficiently, computationally hard. For public-key encryption Dodis *et al.* [35] build schemes based on LWE and DDH (where their DDH-based scheme is a variant of [20]) secure against *subexponentially-hard-to-invert* functions. Brakerski and Goldwasser [23] present schemes with the same level of auxiliary-input security under the subgroup indistinguishability assumption.

1.3 Main Results of the Thesis

As alluded to earlier, all constructions of circular-secure encryption in the literature rely on specific assumptions. Moreover, the current state of knowledge regarding what assumptions are too weak to yield a provable construction of circular security is limited.

The results presented in this thesis are contributions to a general program aimed at understanding the complexity of the notion of circular security. Specifically, we study the notion of circular security from three different angles; namely, in terms of (A) assumptions sufficient to realize this notion, (B) the black-box complexity of assumptions behind this notion and (C) applications enabled by this notion. The results in Chapter 2, 3 and 4 are, respectively, contributions to Points (A), (C) and (B). In what follows we give a gentle exposition of the results contained in each

chapter. We refer the reader to the respective chapter for a more comprehensive introduction, discussing in depth the related work and the significance of the results.

1.3.1 Overview of Results in Chapter 2

In Chapter 2 we build a public-key bit-encryption scheme that provides projection security as well as two forms of key-leakage resilience, from any semantically-secure public-key bit encryption scheme that provides the following two properties:

- a form of group homomorphism with respect to both plaintext messages and randomness; and
- a reproducibility property, stating that given a ciphertext $E_{pk_1}(m_1; r_1)$, message m_2 and a pair of public/secret keys (pk_2, sk_2), one can efficiently compute $E_{pk_2}(m_2; r_1)$, i.e., the randomness underlying a given ciphertext can be reused to encrypt under a new public key whose secret key is known, without knowing anything beyond the public key of the given ciphertext.

We prove that the constructed scheme provides projection security, key-leakage resilience with a $(1 - o(1))$ resilience rate and auxiliary-input security against sub-exponentially-hard functions. Our results explain and unify those of Boneh *et al.* [20] and Brakerski and Goldwasser [23] as well as the leakage resilience properties proved for these schemes [75, 35, 23], and give rise to the first circular-secure scheme based on homomorphic-hash-proof systems [33].

The results presented in this chapter are a subset of those that appeared in [59].

1.3.2 Overview of Results in Chapter 3

In Chapter 3 we give generic constructions of several fundamental cryptographic primitives based on a new encryption notion that combines circular security with reproducibility. Our main results in this chapter are as follows.

- We give a novel generic construction of trapdoor functions (TDFs) from any reproducible public-key bit-encryption scheme and show that successively stronger forms of circular security result in increasingly stronger forms of one-wayness for the TDF. In particular, we quantity the notion of circular security, in terms of the number of (independent) copies of the encrypted key available to the adversary, to obtain a hierarchy of notions that we call k-rec circular security.

We then show, via a tight reduction, that if the base scheme is k-rec circular secure the constructed TDF is k-wise one-way, in the sense of [88].

- We show methods for extracting many hardcore bits (via tight reductions) for the constructed TDF and by applying the result of [88] we obtain a black-box construction of CCA2-secure encryption.
- By slightly extending our base primitive, we show how to obtain deterministic encryption schemes secure under *block-source* inputs, as defined by [18].
- We realize our base encryption primitive by showing that the circular-secure schemes of [20, 23] are reproducible.

Informally speaking, a trapdoor function family is k-wise one-way if the one-wayness holds even when the given input is evaluated under k independently chosen functions from the family. Rosen and Segev [88] show the utility of this notion by giving a black-box construction of CCA-secure encryption from k-wise one-way TDFs, for sufficiently large k. They also show how to obtain k-wise one-way TDFs based on lossy trapdoor functions [83]. This is virtually the only known construction of k-wise one-way TDFs in the literature besides our work. Indeed, Vahlis [92] shows that trapdoor permutations are not sufficient to obtain even 2-wise one-way TDFs in a black-box way.

The results in this chapter are based on [57].

1.3.3 Overview of Results in Chapter 4

In Chapter 4 we investigate the question of whether circular-secure bit encryption can be based on the minimal primitive of semantically-secure encryption. We first show that the related notion of seed-circular-secure bit encryption cannot be based on semantically-secure encryption in a fully-black-box way. This new notion is a variant of the circular-security notion in the sense that the seed (i.e., randomness) of the key generation algorithm, instead of the secret key, is encrypted (under the corresponding public key). Our separation model follows that of Gertner, Malkin and Reingold [48], which only rules out fully-black-box reductions, as opposed to the earlier model of Impagliazzo and Rudich that rules out bigger classes of reductions (e.g., relativizing reductions).

Next, we show how to extend our results to the case of circular security by ruling out a large and non-trivial class of constructions that we call *key-isolating* constructions.

The results in this chapter are based on [58].

1.4 Organization of the Thesis

In the next section we present some notation and conventions that will be used throughout the thesis. The next three chapters present the results mentioned above. Except in one or two places where we may refer to a basic definition discussed in a previous chapter all chapters are independent and can be read in any order. We end each chapter by discussing a list of open problems related to the results of that chapter.

1.5 Notation and Conventions in the Thesis

In this section we fix some notation and conventions that will be used throughout the thesis. Our notational style is fairly standard.

Bitstrings. For $x \in \{0,1\}^*$ we use $|x|$ to denote the bit length of x and use x_i, for $1 \leq i \leq |x|$, to denote the ith bit of x.

Cartesian product. We denote the n-th Cartesian power of a set S by S^n.

Distributions and sampling. For a finite set S we use $x \leftarrow S$ to denote sampling x uniformly at random from S and denote by $Unif_S$ the uniform distribution on S. If $\mathcal{D}$ is a distribution then $x \leftarrow \mathcal{D}$ denotes choosing x according to $\mathcal{D}$; also $x \leftarrow \mathcal{D}^t$ denotes the t-tuple formed by sampling t times independently from $\mathcal{D}$. We denote the support set of a distribution $\mathcal{D}$ by $Sup(\mathcal{D})$, and write $x \in \mathcal{D}$ to indicate $x \in Sup(\mathcal{D})$.

Negligible functions. We call $f : \mathbb{N} \to \mathbb{R}$ *negligible* if $f(n) < 1/P(n)$, for any polynomial P and sufficiently large n. We write *negl* to denote unspecified negligible functions.

Convention about PPT algorithms. We use the term PPT in the standard sense. We will often omit the adjective PPT/efficient when discussing algorithms (honest or adversarial)/distributions – by default we assume all these entities are efficient/efficiently samplable.

Indistinguishability notions. For two ensembles $X = \{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ of random variables we say X is *computationally indistinguishable* from Y, denoted $X \equiv^c Y$, if for any bit-valued, PPT algorithm D, we have $|\Pr[D(X_n) = 1] - \Pr[D(Y_n) = 1]| = negl(n)$. We say X and Y are statistically indistinguishable, denoted by $X \equiv^s Y$, if $f(n) = \Delta[X_n, Y_n]$ is negligible, where Δ denotes statistical distance. We write $X \equiv Y$ to mean X and Y are identically distributed.

Randomized algorithms. We use $A(\ldots; r)$ to denote the deterministic output of a randomized algorithm A when the randomness is fixed to r, and use $x \leftarrow A(a_1, a_2, \ldots)$ to denote the distribution formed by outputting $A(a_1, a_2, \ldots; r)$ for a uniformly-random r.

Chapter 2

Construction of Circular-Secure and Leakage-Resilient Encryption

In this chapter we focus on the first goal related to the program of understanding the complexity of circular security: that is, what assumptions are sufficient for circular security?

We give a generic construction that transforms any public-key bit-encryption scheme with two specific properties into a public-key bit-encryption scheme that satisfies basic forms of KDM security as well as two forms of key-leakage resiliency. Our construction can be used as a basis to unify two existing specific circular-secure constructions [20, 23], to explain their demonstrated leakage-resilience properties [35, 75, 23] and to obtain a circular-secure scheme based on homomorphic hash-proof systems [33]. We, however, only present the general construction and refer the interested reader to [59] for instantiation results. We also mention that the recent work of Wee [94] shows a direct way of constructing circular-secure schemes from homomorphic hash proof systems.

2.1 Chapter Organization

In Section 2.2 we give all the underlying definitions, including those of security for the constructed scheme, as well as those related to the properties of the base scheme. In Section 2.3 we present the construction and in the next three sections we prove the claimed security properties of the constructed scheme. Finally, in Section 2.7 we conclude with a few open problems.

2.2 Definitions

As a convention all groups discussed in this chapter are assumed to admit efficient group operations, and to be commutative, but not necessarily cyclic, unless otherwise indicated.

2.2.1 Syntax of Encryption Schemes

A *public-key bit-encryption scheme* $\mathcal{E}$ is given by algorithms (G, E, Dec), all taking as input a *security parameter* 1^n (that we make explicit for G and implicit for other algorithms). The *key-generation* algorithm, takes 1^n and outputs public/secret keys, $(pk, sk) \leftarrow G(1^n)$. The *encryption* algorithm E, takes a public key pk, a plaintext bit b and randomness $r \in \mathsf{R}_n$ (where R_n is the *randomness space*), and deterministically produces ciphertext $c = E_{pk}(b; r)$. Finally, the *decryption* algorithm takes a secret key sk and ciphertext c, and deterministically outputs $b = Dec_{sk}(c)$. For correctness, we require, for every $(pk, sk) \in G(1^n)$, every $b \in \{0, 1\}$ and $c \in E_{pk}(b)$, that $Dec_{sk}(E_{pk}(b)) = b$. We typically use PK_n and SK_n to refer to the public-key and secret-key spaces. Formally, $(\mathsf{PK}_n, \mathsf{SK}_n) = Sup(G(1^n))$. All encryption schemes in this chapter, if not otherwise stated, are bit-encryption schemes.

2.2.2 Key-Dependent-Message Security

We describe the notion of KDM$^{(t)}$ security for an encryption scheme $\mathcal{E} = (G, E, Dec)$.

KDM security. Assume that $F = \{F_n\}_{n\in\mathbb{N}}$ is an ensemble of sets of functions, where for each $f \in F_n$, it holds that $f : \mathsf{SK}_n^t \to \{0, 1\}$.

We define F-KDM$^{(t)}$ security through the following F-KDM$^{(t)}$ game, played between a challenger and an adversary. The challenger first chooses $b \leftarrow \{0, 1\}$, generates $(pk_1, sk_1), \ldots, (pk_t, sk_t) \leftarrow G(1^n)$, and gives $pk_1, \ldots, pk_t$ to the adversary. The adversary $\mathcal{A}$, given pk_i's, can repeatedly and adaptively, for $1 \le i \le t$, make queries of the form (i, f), where $f \in F_n$, or of the form (i, m), where $m \in \{0, 1\}$, and in return,

- If $b = 0$, the challenger returns $E_{pk_i}(f(sk_1, \ldots, sk_n))$ in response to (i, f) and $E_{pk_i}(m)$ in response to (i, m); and
- If $b = 1$, the challenger returns $E_{pk_i}(0)$.

$\mathcal{A}$ finally outputs a bit b'. We define the F-KDM$^{(t)}$ *advantage of* $\mathcal{A}$ as

$$Adv^{F\text{-KDM}^{(t)}}(\mathcal{A}) = \left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|,$$

where the probabilities are taken over the random coins of $\mathcal{A}$ and of the challenger.

We say that $\mathcal{E}$ is F-KDM$^{(t)}$-secure if for any $\mathcal{A}$ in the above game, it holds that $Adv^{F\text{-KDM}^{(t)}}(\mathcal{A}) = negl$. (Recall by the convention made in Section 1.5 that $\mathcal{A}$ is PPT.)

Circular and projection security. Assume $\mathsf{SK}_n = \{0,1\}^{l(n)}$ and let $l = l(n)$. For $1 \le i \le t$ and $1 \le j \le l$, define $Sel_{i,j} \colon \mathsf{SK}_n^t \mapsto \{0,1\}$ to be the function that on input $(sk_1, \ldots, sk_t)$ returns the jth bit of sk_i. Similarly, define $NSel_{i,j}$ to be the function that on input $(sk_1, \ldots, sk_t)$ returns the negation of the jth bit of sk_i. Finally, define $S_n = \{Sel_{i,j} \colon 1 \le i \le t, 1 \le j \le l\}$ and $\hat{S}_n = \{NSel_{i,j} \colon 1 \le i \le n, 1 \le j \le l\}$. We now give the following definitions.

- We call $\mathcal{E}$ *t-circular secure* if $\mathcal{E}$ is F-KDM$^{(t)}$ secure, where $F_n = S_n$.
- We call $\mathcal{E}$ *t-projection secure* if $\mathcal{E}$ is F-KDM$^{(t)}$ secure for $F_n = S_n \cup \hat{S}_n$.

2.2.3 Leakage Resilience

We define the notion of leakage resilience. For $\pounds = \pounds(n)$, we say that the public-key encryption scheme $\mathcal{E} = (G, E, Dec)$ is *$\pounds$-length leakage resilient* if, for any adversary $\mathcal{A}$, the $\pounds$-leakage-advantage of $\mathcal{A}$, $Adv^{\pounds\text{-}leak}(\mathcal{A})$, defined via the following game, is negligible.

- Setup: The challenger generates $(pk, sk) \leftarrow G(1^n)$ and gives pk to $\mathcal{A}$.
- Leakage queries: $\mathcal{A}$ sends function $f : \mathsf{SK}_n \to \{0,1\}^*$ to the challenger, where $|f(sk)| \le \pounds$, and receives, in response, $f(sk)$.
- Challenge: $\mathcal{A}$ submits $(m_0, m_1) \in \{0,1\}^2$, and the challenger, samples $b \leftarrow \{0,1\}$, and returns $E_{pk}(m_b)$ to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a bit b'.

We define $Adv^{\pounds\text{-}leak}(\mathcal{A}) = \left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|$. We say that $\mathcal{E}$ is *r-rate leakage resilient* (or has *resilience rate* r) if $\mathcal{E}$ is $r \cdot \log|\mathsf{SK}|$-length leakage resilient.

Finally, we note that restricting $\mathcal{A}$ in the above game to a single leakage query is without loss of generality. In particular, the security definition does not become

stronger if $\mathcal{A}$ is allowed to adaptively make multiple leakage queries provided that the total length of the bits leaked is bounded by $\pounds(n)$. The proof of this fact is straightforward; see [1] for a proof.

2.2.4 Auxiliary-Input Security

Let $\mathcal{E} = (G, E, Dec)$ be an encryption scheme with public-key, secret-key and message spaces, respectively, PK_λ, SK_λ and M_λ. Throughout this Section we use f to refer to a function with domain $(\mathsf{PK}_\lambda, \mathsf{SK}_\lambda)$ and range SK_λ. We follow the notation of [23]. For $\mathcal{E} = (G, E, Dec)$ we define *f-weak inversion* and *f-strong inversion* as follows. We say that f is *ϵ-strongly-uninvertible* under $\mathcal{E}$ if for any adversary $\mathcal{A}$, the probability that $\mathcal{A}$ outputs sk when given $(f(pk, sk), pk)$ is at most $\epsilon(\lambda)$, where the probability is taken over $\mathcal{A}$'s random coins and $(pk, sk) \leftarrow G(1^\lambda)$. Also, we say that f is *ϵ-weakly-uninvertible* under $\mathcal{E}$ if for any adversary $\mathcal{A}$, the probability that $\mathcal{A}$ outputs sk when given $f(pk, sk)$ is at most $\epsilon(\lambda)$, where the probability is taken over $\mathcal{A}$'s random coins and $(pk, sk) \leftarrow G(1^\lambda)$. Let Aux_ϵ^{st} be the class of all ϵ-strongly-uninvertible functions and Aux_ϵ^{wk} be the class of all ϵ-weakly-uninvertible functions. Note that $Aux_\epsilon^{st} \subseteq Aux_\epsilon^{wk}$.

We say that $\mathcal{E}$ is *f-auxiliary-input secure* if any adversary $\mathcal{A}$ has a negligible advantage in the following game: $\mathcal{A}$ is given $(pk, f(pk, sk))$, where $(pk, sk) \leftarrow G(1^\lambda)$; $\mathcal{A}$ submits $(m_0, m_1) \in \mathsf{M}_\lambda^2$; $\mathcal{A}$ receives $E_{pk}(m_b)$, for $b \leftarrow \{0, 1\}$; finally, $\mathcal{A}$ outputs bit b', and achieves the following advantage

$$\left|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]\right|.$$

We say that $\mathcal{E}$ is *ϵ-weakly-auxiliary-input secure* (resp., ϵ-strongly-auxiliary-input secure) if $\mathcal{E}$ is *f-auxiliary-input secure* for any $f \in Aux_\epsilon^{st}$ (resp., Aux_ϵ^{wk}). We say $\mathcal{E}$ is auxiliary-input secure against *subexponentially-hard functions* if for some $c > 0$, $\mathcal{E}$ is $1/(2^{\lambda^c})$-strongly-auxiliary-input secure.

2.2.5 Properties of the Base Scheme

We give the definitions of the main properties that we need from the base scheme.

We now review the notion of *reproducibility* of an encryption scheme, as defined in [11].

Definition 1. *We call* $\mathcal{E} = (Gen, E, Dec)$ reproducible *if there exists a deterministic function* R*, called the reproduction function, such that for any* n*, any* (pk_1, sk_1)*,* $(pk_2, sk_2) \in Gen(1^n)$*, any* $r \in Rand_n$ *and any* $b_1, b_2 \in \{0,1\}$*,*

$$R\left(pk_1, E_{pk_1}(b_1; r), b_2, pk_2, sk_2\right) = E_{pk_2}(b_2; r).$$

For simplicity, we omit the inclusion of pk_1 *and* pk_2 *as inputs to* R*.*

We define a strong notion of homomorphism for a general encryption scheme.

Definition 2. *Let* $\mathcal{E} = (G, E, Dec)$ *be a public-key encryption scheme where both* $(\mathsf{R}_n, +)$ *and* $(\mathsf{M}_n, +)$ *form groups. (*M_n *denotes* $\mathcal{E}$*'s plaintext space.) Then* $\mathcal{E}$ *is* additively homomorphic with respect to plaintexts and randomness (PR-additively homomorphic) *if there is an efficient function* Hom *such that for every* $(pk, *) \in G(1^n)$*,* $m_1, m_2 \in \mathsf{M}_n$*, and* $r_1, r_2 \in \mathsf{R}_n$*,*

$$Hom\left(pk, E_{pk}(m_1; r_1), E_{pk}(m_2; r_2)\right) = E_{pk}(m_1 + m_2; r_1 + r_2).$$

We extend the notation of $Hom(\cdot)$ *to define* $Hom(pk, c_1, \ldots, c_m)$ *in the straightforward way. Whenever* pk *is clear from the context we omit the inclusion of* pk *as an input to the* Hom *algorithm.*

2.3 Construction

We first fix some notation. Throughout this section we will be working with additive notation for groups with 0 denoting the identity element. For $\mathbf{g} = (g_1, \ldots, g_p) \in \mathbb{G}^p$ and $\mathbf{b} = (b_1, \ldots, b_p) \in \{0,1\}^p$ we define $\mathbf{b} \cdot \mathbf{g} = b_1 \cdot g_1 + \cdots + b_p \cdot g_p \in \mathbb{G}$, where, $0 \cdot g = 0$, and for $n \in \mathbb{N}$, we define $n \cdot g = g + (n-1) \cdot g$.

We present a generic construction that transforms a reproducible, homomorphic public-key encryption scheme into a public-key bit-encryption scheme. Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure public-key bit-encryption scheme providing reproducibility (with the associated function Rep) and homomorphism (with the associated function Hom). Recall for homomorphism, both the message space, $\{0,1\}$, and the randomness space, R_n, form groups, which implies the plaintext group is just $\mathbb{Z}_2$. We now present the construction.

Construction 1. *(Single bit encryption): Let* $\mathcal{E} = (G, E, Dec, Hom, Rep)$ *be as above and let* $l = l(n)$ *be a value that we instantiate later.*

- *Key generation G': Choose the secret key as $\mathbf{s} \leftarrow \{0,1\}^l$ and the public key as $(E_{pk}(0;r_1),\ldots,E_{pk}(0;r_l),E_{pk}(0;\mathbf{s}\cdot\mathbf{r}))$, where $(pk,*) \leftarrow G(1^n)$, $r_1,\ldots,r_l \leftarrow \mathsf{R}_n$ and $\mathbf{r}=(r_1,\ldots,r_l)$.*

- *Encryption E': To encrypt bit b under public key $(c_1,\ldots,c_l,c_{l+1})$, do the following: choose $(pk',sk') \leftarrow G(1^n)$ and return $(c'_1,\ldots,c'_l,c'_{l+1})$, where $c'_i = Rep(c_i,0,sk')$, for $1 \le i \le l$, and $c'_{l+1} = Rep(c_{l+1},b,sk')$.*

- *Decryption Dec': To decrypt $(c'_1,\ldots,c'_l,c'_{l+1})$ under secret key $\mathbf{s}$, letting $(i_1,\ldots,i_w)$ be the indices of non-zero bits of s, output 0 if $c'_{l+1} = Hom\left(c'_{i_1},\ldots,c'_{i_w}\right)$, and 1 otherwise.*

The completeness of the scheme follows immediately. A few comments are in order. First, the encryption algorithm of the constructed scheme uses that of the base scheme, but by reusing the randomness values of the ciphertexts given in the public key. Second, the constructed decryption function does not need any secret keys of the base scheme, e.g., sk, for its computation. Roughly, this is why proving circular security for the constructed scheme should not be much harder than proving CPA security. In our security proofs, we will rely on the fact that we may use the homomorphism properties of the base primitive to form public keys and encryptions in alternate, equivalent ways as described below.

Proposition 1. *1. The public key may be computed as $(c_1,\ldots,c_l,c_{l+1})$, where $c_i \leftarrow E_{pk}(0)$, for $1 \le i \le l$, and $c_{l+1} = Hom\,(c_{i_1},\ldots,c_{i_w})$, where $(i_1,\ldots,i_w)$ are the indices of non-zero bits of $\mathbf{s}$.*

2. Let $\mathbf{s}$, sk' and $c'_1,\ldots,c'_l$ be as in the definition of encryption in Construction 1. Then, c'_{l+1} may be computed as $c'_{l+1} = Hom(c_{i_1},\ldots,c_{i_w})$, where $(i_1,\ldots,i_w)$ are the indices of non-zero bits of $\mathbf{s}$.

2.4 Proof of Projection Security

In this section we give the proof of projection security of our constructed scheme. For clarity of exposition, we present the results with respect to asymptotic security, without specifying the exact advantage functions. This section is organized as follows. In Subsection 2.4.1 we reviews some facts related to entropy which are needed by our proofs. In Subsection 2.4.2 we introduce an intermediate lemma that will be used in

the proofs of our main theorems. Finally, in Subsection 2.4.3 we give the proof for projection security.

2.4.1 Information-Theoretic Tools

We denote the *min-entropy* of a distribution D by $H_\infty(D)$, defined as $H_\infty(D) = \min_{d \in D}\left[\log(\frac{1}{\Pr[D=d]})\right]$. We also need to work with the notion of *average min entropy*, formalized by Dodis *et al.* [37], which measures the expected unpredictability of X given a random value y of Y. Formally,

$$\widetilde{H}_\infty(X|Y) = -\log\left(E_{y\leftarrow Y}(2^{-H_\infty(X|Y=y)})\right) = -\log\left(E_{y\leftarrow Y}(\max_x \Pr[X=x|Y=y])\right).$$

A well-known fact about average-min entropy is a special form of the *chain rule*, saying that conditioning on a random variable Y, the average min entropy decreases by at most the logarithm of the support size of Y.

Lemma 1. *([37]) For any (X, Y, Z) it holds that $\widetilde{H}_\infty(X|Y,Z) \geq \widetilde{H}_\infty(X|Z) - \log|Sup(Y)|$.*

A family of functions $\{h : D \to R\}_{h\in H}$ is called *universal* if for all $x_1, x_2 \in D$, with $x_1 \neq x_2$, it holds that

$$\Pr_{h\leftarrow H}[h(x_1) = h(x_2)] \leq \frac{1}{|R|}.$$

We typically denote a family of functions $\{h : D \to R\}_{h\in H}$ as a single function $\mathcal{H} : D \times H \to R$, where $\mathcal{H}(d,h) = h(d)$. We have the following fact, showing that universal hash functions are good *average-case extractors*.

Lemma 2. *([37]) If $Ext : \{0,1\}^k \times W \to W'$ is a family of universal hash functions, then for any pair of random variables (D, X), where D takes values in $\{0,1\}^k$, it holds that*

$$\Delta\left((Ext(D,S),S,X),(R,S,X)\right) \leq 1/2\sqrt{2^{-\widetilde{H}_\infty(D|X)}|W'|},$$

where S is uniform over W, R is uniform over W' and Δ denotes statistical distance. We stress that S is independent of (D, X).

2.4.2 A Useful Lemma

We begin by introducing a game that will be used in proving our main results. Intuitively, the following experiment corresponds to a vector-encryption game, in which

an adversary may interactively issue vectors of bits (of certain forms) to be encrypted, and each vector is component-wise encrypted under a fresh secret key while reusing randomness across each fixed component of vectors.

The randomness-sharing (RS) Game. Let (G, E, Dec) be a public-key bit-encryption scheme. As some notation, for $l \in \mathbb{N}$, we let $\mathbf{e}_i^l$, for $1 \leq i \leq l$, be the the vector of size l which has 1 in the ith position and 0 everywhere else, and $\mathbf{e'}_i^l$, for $1 \leq i \leq l$, be the vector of size l which has 1 in both its ith position and last position, and 0 everywhere else. We let $\mathbf{0}^l$ be the all-0 vector of size l. Finally, for $\mathbf{b} = (b_1, \ldots, b_l)$ and $\mathbf{r} = (r_1, \ldots, r_l)$, we define $E_{pk}(\mathbf{b}; \mathbf{r}) = (E_{pk}(b_1; r_1), \ldots, E_{pk}(b_l; r_l))$.

The game is parameterized over $l = l(n)$ and $t = t(n)$ and is played as follows.

The challenger chooses $b \leftarrow \{0, 1\}$ and for each $1 \leq h \leq t$ it samples $\mathbf{r}_h = (r_{h1}, \ldots, r_{hl}) \leftarrow \mathsf{R}_n^l$. Then the game proceeds as follows: the adversary repeatedly and adaptively makes queries of the form $(h, \mathbf{e})$, where $1 \leq h \leq t$ and $\mathbf{e} \in \{\mathbf{0}^l\} \cup \{\mathbf{e}_1^l, \ldots, \mathbf{e}_l^l\} \cup \{\mathbf{e'}_1^l, \ldots, \mathbf{e'}_l^l\}$, and in response to each such query, the challenger samples $(pk, *) \leftarrow G(1^n)$ (using fresh coins for each query) and returns $(pk, E_{pk}(\mathbf{e}; \mathbf{r}_h))$ if $b = 0$, and $(pk, E_{pk}(\mathbf{0}^l; \mathbf{r}_h))$, otherwise. Finally, the adversary outputs a bit b' and its advantage is defined as:

$$Adv^{\text{p-rs}}(\mathcal{A}) = \Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1].$$

We now give the following lemma.

Lemma 3. *Assume $\mathcal{E} = (G, E, Dec, Rep)$ is a CPA-secure, public-key bit-encryption scheme that provides reproducibility. For any polynomial functions $l(\cdot)$ and $n(\cdot)$, and any adversary $\mathcal{A}$ in the (l, t)-RS game has a negligible advantage.*

Proof. For convenience we drop the security parameter from all sets throughout the proof. Using a simple hybrid argument we can show that any adversary that has advantage ϵ against the (l, t)-RS game can be reduced to an adversary with advantage ϵ/t against the $(l, 1)$-RS game. So we assume $t = 1$.

First, we introduce the following notation. For $\mathbf{b} = (b_1, \ldots, b_l)$ and $\mathbf{c} = (c_1, \ldots, c_l)$, define

$$Rep(\mathbf{c}, \mathbf{b}, sk) = (Rep(c_1, b_1, sk), \ldots, Rep(c_l, b_l, sk)).$$

Assuming that $\mathcal{A}$ makes $t = t(n)$ queries $\mathbf{q}_1, \ldots, \mathbf{q}_t$ we define the hybrid W_i, for $1 \leq i \leq t + 1$, as follows: first generate randomness vector $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}^l$ and respond to queries as follows: in response to the j'th query, for $1 \leq j < i$, generate

$(pk_j, sk_j) \leftarrow G(1^n)$ and return $(pk_j, E_{pk_j}(\mathbf{q}_j; \mathbf{r}))$ (i.e., encryption of the actual vector); and in response to the w'th query, for $w \geq i$, generate $(pk_w, sk_w) \leftarrow G(1^n)$ and return $(pk_w, E_{pk_w}(\mathbf{0}^l; \mathbf{r}))$ (i.e., encryption of the all-zero vector). Note that W_1 and W_{t+1} match exactly the view of the adversary produced under the the RS game when $b = 1$ and $b = 0$, respectively. Thus, for the rest of the proof, we show how to reduce an adversary that can distinguish between W_i and W_{i+1}, for some $1 \leq i \leq t$, to an adversary against the CPA security game; the whole proof then follows using a standard hybrid argument.

Assume that $\mathcal{A}'$ can distinguish between W_i and W_{i+1} with a non-negligible advantage. Noting that W_i and W_{i+1} only differ in the way that the answer to the ith query is made, and that each query vector can take at most $2l+1$ different values, we guess the ith query vector (that is going to be issued by $\mathcal{A}'$), call the LOR-CPA oracle, which is parameterized over a known public key pk, on the guessed vector to receive $\mathbf{c} = (c_1, \ldots, c_l)$, and start simulating $\mathcal{A}'$ as follows: in response to the j'th query, $\mathbf{q}_j$, for $1 \leq j < i$, we generate $(pk_j, sk_j) \leftarrow G(1^n)$ and return $(pk_j, Rep(\mathbf{c}, \mathbf{q}_j, sk_j))$; in response to the ith query we return $(pk, \mathbf{c})$ (if our guess for $\mathbf{q}_i$ was incorrect, we stop and return a random bit); and in response to the w'th query, $\mathbf{q}_w$, for $w > i$, we generate $(pk_w, sk_w) \leftarrow G(1^n)$ and return $(pk_w, Rep(\mathbf{c}, \mathbf{0}^l, sk_w))$. Now it is easy to see that, if our guessing for the ith query was correct, depending on whether the CPA-challenge bit was zero or one, the resulting experiment matches exactly either W_i or W_{i+1}. This completes the proof. □

2.4.3 Proof of Projection Security

We first give the proof of 1-projection security of our scheme and then present a proof for n-projection security. Our proofs build on techniques from [23], which in turn generalize the DDH-based techniques of [20].

Theorem 1. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure public-key bit-encryption scheme providing PR-additive homomorphism and reproducibility.Then, by taking $l = l(n) = \omega(\log n) + \log(|\mathsf{R}_n|)$, the scheme built in Construction 1 is 1-projection secure.*

Proof. To represent the 1-projection game more concisely, we denote:

- *enc-secret*(i) encrypt the ith bit of the secret key; and
- *enc-secret*$(\bar{i})$ encrypt the negation of the ith bit of the secret key.

We introduce a series of hybrid games and show no adversary can distinguish between any two adjacent games. The first game corresponds to the real-encryption circular-security game, while the last game is the one where we always encrypt 0. Letting x_i be the adversary's output in *Game-i*, we write *Game-i* $\equiv^G$ *Game-j* to indicate $|\Pr[x_i = 1] - \Pr[x_j = 1]| = negl$. In all these games, whenever we write, say, $(pk', sk') \leftarrow G(1^n)$ we mean that (pk', sk') is chosen freshly, so we may keep using the same variable sk' inside each game whenever we are producing a new key. Let $\mathsf{R} = \mathsf{R}_n$ for the following discussion. Also, recall the notation $E_{pk}(\mathbf{b}, \mathbf{r})$ introduced in Subsection 2.4.2. Below we write $\mathbf{e}_i$ as shorthand for $\mathbf{e}_i^l$.

Game-0: real encryption. This game provides the adversary with a view that is identical to that under the projection security game in which the challenge bit is zero. The identical view is produced by using the algorithm Hom to produce the public key and to reply to encryption queries. (See Proposition 1.)

Generate $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}^l$ and $\mathbf{s} \leftarrow \{0,1\}^l$ and let $(i_1, \ldots, i_w)$ be the indices of nonzero bits of $\mathbf{s}$. Then,

- the adversary is given $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}))$ as the public key, where

  $$(c_1, \ldots, c_l) = E_{pk}(\mathbf{0}^l; \mathbf{r})$$

  and $(pk, sk) \leftarrow G(1^n)$.

- In response to *enc-secret*(i) we return $(c'_1, \ldots, c'_l, Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{pk'}(\mathbf{s}_i; 0)))$, where

  $$(c'_1, \ldots, c'_l) = E_{pk'}(\mathbf{0}^l; \mathbf{r})$$

  and $(pk', sk') \leftarrow G(1^n)$. Again we emphasize sk' is chosen freshly for each query.

- In response to *enc-secret*$(\bar{i})$ we return $(c''_1, \ldots, c''_l, Hom(c''_{i_1}, \ldots, c''_{i_w}, E_{pk''}(\bar{\mathbf{s}}_i; 0)))$, where

  $$(c''_1, \ldots, c''_l) = E_{pk''}(\mathbf{0}^l; \mathbf{r})$$

  and $(pk'', sk'') \leftarrow G(1^n)$.

Game-1: In this game we handle key generation exactly as in *Game*-0, but we reply to *enc-secret* queries in a special way. Formally, generate $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}^l$ and $\mathbf{s} \leftarrow \{0,1\}^l$ and let $(i_1, \ldots, i_w)$ be the indices of nonzero bits of $\mathbf{s}$. Then,

- the adversary is given $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}))$ as the public key, where $(c_1, \ldots, c_l) = E_{pk}(\mathbf{0}^l; \mathbf{r})$, for $(pk, sk) \leftarrow G(1^n)$.
- In response to *enc-secret*(i) we return $(c'_1, \ldots, c'_l, Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{pk'}(\mathbf{s}_i; 0)))$, where $(c'_1, \ldots, c'_l) = E_{pk'}(\mathbf{e}_i; \mathbf{r})$ and $(pk', sk') \leftarrow G(1^n)$.
- In response to *enc-secret*$(\bar{i})$ we return $(c''_1, \ldots, c''_l, Hom(c''_{i_1}, \ldots, c''_{i_w}, E_{pk''}(\bar{\mathbf{s}}_i; 0)))$, where $(c''_1, \ldots, c''_l) = E_{pk''}(\mathbf{e}_i; \mathbf{r})$ and $(pk'', sk'') \leftarrow G(1^n)$.

We claim that the difference between *Game*-0 and *Game*-1 can be simulated through the l-RS game. The reason is if we know $\mathbf{s}$, then we can compute

$$Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{pk'}(\mathbf{s}_i; 0))$$

from $(c'_1, \ldots, c'_l)$. A similar argument holds with respect to c and c''. Moreover, for every $1 \leq j \leq l$, the ciphertexts c_j, c'_j and c''_j were formed under the same randomness. Thus, we can reduce any distinguisher between *Game*-0 and *Game*-1 to an l-RS game adversary $\mathcal{A}$ as follows: $\mathcal{A}$ samples $\mathbf{s} \leftarrow \{0,1\}^l$ and lets $(i_1, \ldots, i_w)$ be the indices of nonzero bits of $\mathbf{s}$; it calls its RS-oracle on $\mathbf{0}^l$ to receive $(c_1, \ldots, c_l)$ and then returns $(c_1, \ldots, c_l, Hom(c_{i_1}, \ldots, c_{i_w}))$ as the public key; it responds to *enc-secret*(i) by first calling its oracle on $\mathbf{e}_i$ to get $(c'_1, \ldots, c'_l)$ and then returning

$$(c'_1, \ldots, c'_l, Hom(c'_{i_1}, \ldots, c'_{i_w}, E_{pk'}(\mathbf{s}_i; 0)));$$

it responds to *enc-secret*$(\bar{i})$ in a similar way. Thus, by Lemma 3 we obtain that *Game*-0 $\equiv^G$ *Game*-1.

Finally, note that under this game, the distribution of the public key and the distributions of responses to *enc-secret*(i)'s and to *enc-secret*$(\bar{i})$'s are:

$$\begin{array}{ll} \big(E_{pk}(\mathbf{0}^l; \mathbf{r}), E_{pk}(0; r_{l+1})\big) & \text{public key} \\ (E_{pk}(\mathbf{e}_i; \mathbf{r}), E_{pk'}(0; r_{l+1})) & \textit{enc-secret}(i) \\ (E_{pk}(\mathbf{e}_i; \mathbf{r}), E_{pk''}(1; r_{l+1})) & \textit{enc-secret}(\bar{i}), \end{array} \tag{2.4.1}$$

where $(pk, *), (pk', *), (pk'', *) \leftarrow G(1^n)$, $\mathbf{s} \leftarrow \{0,1\}^l$ and $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}^l$ and $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$. In particular, note that the bits of $\mathbf{s}$ never appear as a plaintext (under E) in Equation 2.4.1, and the only place we use $\mathbf{s}$ is to form r_{l+1}.

Game-2: This game proceeds exactly as in *Game*-1, except we now sample r_{l+1} independently of all other r_i's. Namely, we sample $(r_1, \ldots, r_l, r_{l+1}) \leftarrow \mathsf{R}^{l+1}$ and run the game by forming the public key and responses to the adversary's queries exactly as in Equation 2.4.1. Notice that the entire game can be simulated by only knowing $(r_1, \ldots, r_l, r_{l+1})$: we generate the public key and we answer to *enc-secret* queries by sampling pk, pk' and pk'' on our own and forming the outputs as spelled out by Equation 2.4.1. (Here we are exploiting the fact that the bits of $\mathbf{s}$ never appear as a plaintext under E in Equation 2.4.1.) Thus, since $l = \omega(\log n) + \log(|\mathsf{R}|)$ and the inner product is a family of universal hash functions, by Lemma 2 (indeed by the Leftover Hash Lemma, which is a special case of Lemma 2) we obtain that the statistical distance between $(\mathbf{r}, \mathbf{s} \cdot \mathbf{r})$ and a tuple chosen uniformly at random from R_n^l is at most $\sqrt{1/2^{\omega(\log n)}} = negl(n)$, and thus *Game*-1 $\equiv^G$ *Game*-2.

Game-3: In this game we again sample r_{l+1} independently of other r_i's, but reply to all queries as "encryptions" of zero. That is, we generate $(r_1, \ldots, r_l, r_{l+1}) \leftarrow \mathsf{R}^{l+1}$ and form the public key and responses to the adversary's queries as follows:

$$\begin{array}{ll} (E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l), E_{pk}(0; r_{l+1})) & \text{public key} \\ (E_{pk'}(0; r_1), \ldots, E_{pk'}(0; r_l), E_{pk'}(0; r_{l+1})) & \text{response to all queries} \end{array} \quad (2.4.2)$$

where, again, $(pk', *)$ is sampled freshly for each query. Now using the fact that all r_i's are sampled independently, and also that pk' is generated using fresh coins each time, we obtain that any adversary that can distinguish between *Game*-2 and *Game*-3 can be reduced to break the $(l+1)$-RS security of $\mathcal{E}$ (which is a contradiction by Lemma (3)). Thus, *Game*-2 $\equiv^G$ *Game*-3.

Game-4: In this game we change back the distributions of r_i's to the original, but answer to all the adversary's queries as encryptions of zero. That is, we generate $\mathbf{s} \leftarrow \{0,1\}^l$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}^l$, let $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$, and form the public key and responses to the adversary's queries as follows:

$$\begin{array}{ll} (E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l), E_{pk}(0; r_{l+1})) & \text{public key} \\ (E_{pk'}(0; r_1), \ldots, E_{pk'}(0; r_l), E_{pk'}(0; r_{l+1})) & \text{responses to all queries} \end{array} \quad (2.4.3)$$

Now, similarly to our proof of *Game*-1 $\equiv^G$ *Game*-2, since *Game*-3 and *Game*-4 differ

only in the way that $(r_1, \ldots, r_l, r_{l+1})$ is generated, and again using the fact that $l = \omega(\log n) + \log(|\mathsf{R}|)$, by applying Lemma 2, we conclude that *Game*-3 $\equiv^G$ *Game*-4. This completes the proof. □

We now give the statement and proof for n-projection security.

Theorem 2. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure public-key bit-encryption scheme providing PR-additive homomorphism and reproducibility. For any constant $c > 1$, by taking $l = n \log(|\mathsf{R}_n|) + \omega(\log n)$, the scheme built in Construction 1 is n-projection secure.*

Proof. To represent the n-projection game more concisely, we use the following notation for denoting the adversary's queries.

- $Enc(pub_h, secret_i, j)$ encrypt the jth bit of the ith secret key under the hth public key; and
- $Enc(pub_h, secret_i, \bar{j})$ encrypt the negation of the jth bit of the ith secret key under the hth public key.

We also need the following notation. In all the games below we denote the ith secret key as $\mathbf{s}_i$ and define $\mathbf{s}_{ij}$ to be the jth-bit of the ith secret key. If $\mathbf{v}$ is a vector of size l, and $\mathbf{s} \in \{0,1\}^l$ we define $\mathbf{v}[\mathbf{s}]$ to be $(\mathbf{v}_{i_1}, \ldots, \mathbf{v}_{i_w})$, where $i_1 < \cdots < i_w$ are the indices of nonzero bits of $\mathbf{s}$. We also recall the notation $\mathbf{e}_i^l$, $\mathbf{0}^l$, and $E_{pk}(\mathbf{b}; \mathbf{r})$, defined at the beginning of Section 2.4. To ease notation though, we shall write $\mathbf{e}_i$ to mean $\mathbf{e}_i^l$.

We introduce a series of hybrid games and show no adversary can distinguish between any two adjacent games. The first game corresponds to the real-encryption n-circular-security game, while the last game is the one where we always encrypt 0. Letting x_i be the adversary's output in *Game-i*, we write *Game-i* $\equiv^c$ *Game-j* to indicate $|\Pr[x_i = 1] - \Pr[x_j = 1]| = negl$. In all these games, whenever we write $sk \leftarrow G(1^n)$ we mean that sk is chosen freshly, so we keep using the same variable sk inside each game whenever we are producing a new key.

Game-0: real encryption. This game provides the adversary with a view that is identical to that under the projection security game in which the challenge bit is zero. The identical view is produced by using the algorithm Hom to produce the public keys and to reply to encryption queries. (See the first paragraph following Construction 1.) Generate $\mathbf{s}_1, \ldots, \mathbf{s}_n \leftarrow \{0,1\}^l$ and for $1 \le i \le n$, $\mathbf{r}_i = (r_{i1}, \ldots, r_{il}) \leftarrow \mathsf{R}_n^l$. (Here $\mathbf{s}_i$

will be the secret key of the ith "user" and $\mathbf{r}_i$ will be used to produce the public key for the ith user.) For each h, where $1 \leq h \leq n$, do:

- the adversary is given public key $(\mathbf{c}, Hom(\mathbf{c}[\mathbf{s}_h]))$, where $\mathbf{c} = E_{pk}(\mathbf{0}^l, \mathbf{r}_h)$, for $(pk, *) \leftarrow G(1^n)$. Recall that $E_{pk}(\mathbf{0}^l, \mathbf{r}_h) = (E_{pk}(0; r_{h1}), \ldots, E_{pk}(0; r_{hl}))$.
- In response to $Enc(pub_h, secret_i, j)$, we return $(\mathbf{c}', Hom(\mathbf{c}'[\mathbf{s}_h], E_{pk'}(\mathbf{s}_{ij}; 0)))$, where $(pk', *) \leftarrow G(1^n)$, $\mathbf{c}' = E_{pk'}(\mathbf{0}^l, \mathbf{r}_h) = (E_{pk'}(0; r_{h1}), \ldots, E_{pk'}(0; r_{hl}))$. Recall by Proposition 1 that this provides an identical encryption view. Again we stress that sk' is chosen freshly for each query.
- In response to $Enc(pub_h, secret_i, \bar{j})$, we return $(\mathbf{c}'', Hom(\mathbf{c}''[\mathbf{s}_h], E_{pk''}(\bar{\mathbf{s}}_{ij}; 0)))$, where $(pk'', *) \leftarrow G(1^n)$, $\mathbf{c}'' = E_{pk''}(\mathbf{0}^l, \mathbf{r}_h) = (E_{pk''}(0; r_{h1}), \ldots, E_{pk'}(0; r_{hl}))$.

<u>*Game*-1</u>: In this game we handle key generation exactly as in *Game*-0, but we reply to *Enc* queries in the following way. For each h, where $1 \leq h \leq n$,

- in response to $Enc(pub_h, secret_i, j)$ we return $(\mathbf{c}', Hom(\mathbf{c}'[\mathbf{s}_h], E_{pk'}(\mathbf{s}_{ij}; 0)))$, where $(pk', *) \leftarrow G(1^n)$ and $\mathbf{c}' = E_{pk'}(\mathbf{e}_j; \mathbf{r}_h)$.
- in response to $Enc(pub_h, secret_i, \bar{j})$ we return $(\mathbf{c}'', Hom(\mathbf{c}''[\mathbf{s}_h], E_{pk''}(\bar{\mathbf{s}}_{ij}; 0)))$, where $(pk'', *) \leftarrow G(1^n)$ and $\mathbf{c}'' = E_{pk''}(\mathbf{e}_j; \mathbf{r}_h)$.

We now wish to use Lemma 3 to prove *Game*-0 $\equiv^c$ *Game*-1. First, note that in both games the hth public key is sampled as $(\mathbf{c}, Hom(\mathbf{c}[\mathbf{s}_h]))$, for $\mathbf{c} = E_{pk}(\mathbf{0}^l; \mathbf{r}_h)$. Moreover, in both games, the response to $Enc(pub_h, secret_i, j)$ is formed as

$$(\mathbf{c}', Hom(\mathbf{c}'[\mathbf{s}_h], E_{pk'}(\mathbf{s}_{ij}; 0)))$$

with the only difference that in *Game*-0, $\mathbf{c}'$ are bitwise encryptions of $\mathbf{0}^l$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh pk', while in *Game*-1, $\mathbf{c}'$ are bitwise encryptions of $\mathbf{e}_j$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh pk'. Similarly, in both games, the response to $Enc(pub_h, secret_i, \bar{j})$ is formed as $(\mathbf{c}'', Hom(\mathbf{c}''[\mathbf{s}_h], E_{pk''}(\bar{\mathbf{s}}_{ij}; 0)))$ with the only difference that in *Game*-0, $\mathbf{c}''$ are bitwise encryptions of $\mathbf{0}^l$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh pk'', while in *Game*-1, $\mathbf{c}''$ are bitwise encryptions of $\mathbf{e}_j$ under the randomness values $(r_{h1}, \ldots, r_{hl})$ and under a fresh pk''. Finally, note that if we have all of $\mathbf{s}_1, \ldots, \mathbf{s}_n$ and if we get $(pk', \mathbf{c}')$ from an oracle, knowing that $\mathbf{c}'$ corresponds either to $E_{pk'}(\mathbf{0}^l, \mathbf{r}_h)$

or to $E_{pk'}(\mathbf{e}_j, \mathbf{r}_h)$, then we can compute $Hom(\mathbf{c}'[\mathbf{s}_h], E_{pk'}(\mathbf{s}_{ij}; 0))$. A similar argument holds for $\mathbf{c}''$.

From the discussion above we can see how to reduce a distinguisher between *Game*-0 and *Game*-1 to an adversary $\mathcal{A}$ against the (l, n)-RS security of the public-key scheme $\mathcal{E}$: $\mathcal{A}$ samples $\mathbf{s}_h$, for each $1 \leq h \leq n$ by itself; it generates the hth public key by querying its RS-oracle on $(h, \mathbf{0}^l)$ to get $(pk, \mathbf{c})$ and then outputting $(pk, \mathbf{c}, Hom(\mathbf{c}[\mathbf{s}_h]))$; it responds to an $Enc(pub_h, secret_i, j)$ query by calling its RS-oracle on $(h, \mathbf{e}_j)$ to get $\mathbf{c}'$ and forming the output as explained above; and it responds to an $Enc(pub_h, secret_i, \bar{j})$ in a similar manner.

Finally, note that in *Game*-1, for each $1 \leq h \leq n$, the distributions of the public key, the response to $Enc(pub_h, secret_i, j)$ and the response to $Enc(pub_h, secret_i, \bar{j})$ are indeed identical to:

$$\begin{aligned}
&(E_{pk}(\mathbf{0}^l; \mathbf{r}_h), E_{pk}(0; r_{h,l+1})) \qquad \text{public key} \\
&(E_{pk'}(\mathbf{e}_j^l; \mathbf{r}_h), E_{pk'}(\mathbf{s}_{ij} + \mathbf{s}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, j) \\
&(E_{pk''}(\mathbf{e}_j^l; \mathbf{r}_h), E_{pk''}(\bar{\mathbf{s}}_{ij} + \mathbf{s}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, \bar{j}),
\end{aligned} \tag{2.4.4}$$

where $r_{h,l+1} = \mathbf{s}_h \cdot \mathbf{r}_h$.

Game-2: In this game we generate the distributions as in Equation 2.4.4, but with a small deviation. For $1 \leq h \leq n$ we generate $\mathbf{s}_h \leftarrow \{0,1\}^l$ and $\mathbf{r}_h \leftarrow \mathsf{R}_n^l$, and set $r_{h,l+1} = \mathbf{s}_h \cdot \mathbf{r}_h$. Moreover, for $1 \leq i \leq n$ we set $\mathbf{d}_i = \mathbf{s}_1 + \mathbf{s}_i$, i.e., bitwise binary addition. We form the hth public key and responses to $Enc(pub_h, secret_i, j)$ and to $Enc(pub_h, secret_i, \bar{j})$ queries as follows:

$$\begin{aligned}
&(E_{pk}(\mathbf{0}^l; \mathbf{r}_h), E_{pk}(0; r_{h,l+1})) \qquad \text{public key} \\
&(E_{pk'}(\mathbf{e}_j^l; \mathbf{r}_h), E_{pk'}(\mathbf{d}_{ij} + \mathbf{d}_{hj}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, j) \\
&(E_{pk''}(\mathbf{e}_j^l; \mathbf{r}_h), E_{pk''}(\overline{\mathbf{d}_{ij} + \mathbf{d}_{hj}}; r_{h,l+1})) \qquad \text{response to } Enc(pub_h, secret_i, \bar{j}),
\end{aligned} \tag{2.4.5}$$

Now since $\mathbf{s}_{ij} + \mathbf{s}_{hj} = \mathbf{s}_{1j} + \mathbf{s}_{ij} + \mathbf{s}_{1j} + \mathbf{s}_{hj} = \mathbf{d}_{ij} + \mathbf{d}_{hj}$ and $\bar{\mathbf{s}}_{ij} + \mathbf{s}_{hj} = \mathbf{s}_{1j} + \bar{\mathbf{s}}_{ij} + s_{1j} + s_{hj} = \bar{\mathbf{d}}_{ij} + \mathbf{d}_{hj} = \overline{\mathbf{d}_{ij} + \mathbf{d}_{hj}}$, and since the adversarial view under *Game*-1 is as in Equation 2.4.4, we get that *Game*-1 and *Game*-2 provide identically-distributed adversarial

views. Finally, note that by having only the tuple

$$D = (\mathbf{r}_1, \ldots, \mathbf{r}_n, r_{1,l+1}, \ldots, r_{n,l+1}, \mathbf{d}_2, \ldots, \mathbf{d}_n), \tag{2.4.6}$$

(and in particular without having the individual $\mathbf{s}_1, \ldots, \mathbf{s}_n$) we can perfectly simulate all the distributions given in Equation 2.4.5, by sampling sk, sk' and sk'' by ourselves. Thus, the whole view of an adversary in this game can be simulated by only having D, through a simulation algorithm $Sim()$.

Game-3: This game proceeds exactly as in *Game*-2, to form the view as $Sim(D)$, except that we generate D (given in Equation 2.4.6) differently from *Game*-2 by sampling $r_{h,l+1}$, for all $1 \leq h \leq n$, independently of $\mathbf{r}_h$; the rest of D remains unchanged. Now we show that the following two distributions are statistically indistinguishable, which by the last statement of *Game*-2, it implies that *Game*-2 $\equiv^s$ *Game*-3;

$$\begin{aligned} D &= (\mathbf{r}_1, \ldots, \mathbf{r}_n, \mathbf{s}_1 \cdot \mathbf{r}_1, \ldots, \mathbf{s}_n \cdot \mathbf{r}_n, \mathbf{d}_2, \ldots, \mathbf{d}_n), \\ D' &= (\mathbf{r}_1, \ldots, \mathbf{r}_n, r_1, \ldots, r_n, \mathbf{d}_2, \ldots, \mathbf{d}_n), \end{aligned} \tag{2.4.7}$$

where $\mathbf{r}_1, \ldots, \mathbf{r}_n \leftarrow \mathsf{R}_n^l$, $\mathbf{s}_1, \ldots, \mathbf{s}_n \leftarrow \{0,1\}^l$, and, for $2 \leq i \leq n$, $\mathbf{d}_i = \mathbf{s}_1 + \mathbf{s}_i$. To show Equation 2.4.7, observe that

$$\begin{aligned} \widetilde{H}_\infty(\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n \mid \mathbf{d}_2, \ldots, \mathbf{d}_n) &\geq H_\infty(\mathbf{s}_1, \mathbf{s}_2 \ldots \mathbf{s}_n) - (n-1)l \\ &= l = n \log |\mathsf{R}_n| + \omega(\log n). \end{aligned}$$

Now considering the fact that

$$Ext((\mathbf{s}_1, \ldots, \mathbf{s}_n), (\mathbf{r}_1, \ldots, \mathbf{r}_n)) =^{\text{def}} (\mathbf{s}_1 \cdot \mathbf{r}_1, \ldots, \mathbf{s}_n \cdot \mathbf{r}_n)$$

is a family of universal hash functions, by applying Lemma (2) we obtain that the statistical distance between D and D' is at most $\sqrt{1/2^{\omega(\log n)}}$, which is negligible.

Note that under this game for $1 \leq h \leq n$, the distributions of the public key, the response to $Enc(pub_h, secret_i, j)$ and the response to $Enc(pub_h, secret_i, \bar{j})$ are indeed

identical to:

$$\begin{aligned}
&(E_{pk}(\mathbf{0}^l;\mathbf{r}_h), E_{pk}(0;r_{h,l+1})) && \text{public key}\\
&(E_{pk'}(\mathbf{e}_j^l;\mathbf{r}_h), E_{pk'}(\mathbf{d}_{ij}+\mathbf{d}_{hj};r_{h,l+1})) && \text{response to } Enc(pub_h, secret_i, j)\\
&(E_{pk''}(\mathbf{e}_j^l;\mathbf{r}_h), E_{pk''}(\overline{\mathbf{d}_{ij}+\mathbf{d}_{hj}};r_{h,l+1})) && \text{response to } Enc(pub_h, secret_i, \bar{j}),
\end{aligned} \tag{2.4.8}$$

where $\mathbf{r}_1,\ldots,\mathbf{r}_h \leftarrow \mathsf{R}_n^l$, $r_{1,l+1},\ldots,r_{n,l+1} \leftarrow \mathsf{R}_n$, $\mathbf{s}_1,\ldots,\mathbf{s}_n \leftarrow \{0,1\}^l$ and $\mathbf{d}_i = \mathbf{s}_i + \mathbf{s}_1$, for $1 \le i \le l$. Note that $\mathbf{s}_i$'s, except in forming $\mathbf{d}_i$'s, play no other role in the distributions above.

Game-4: This game proceeds as in *Game*-3 (i.e., Equation 2.4.8), but we reply to all queries as "encryptions" of zero. That is, generate $\mathbf{r}_1,\ldots,\mathbf{r}_n \leftarrow \mathsf{R}_n^n$, and for $1 \le h \le n$, form the hth public key as $(E_{pk}(\mathbf{0}^l;\mathbf{r}_h), E_{pk}(0;r_{h,l+1}))$, where $r_{h,l+1}$ is chosen independently of $\mathbf{r}_h$, and respond to every query under this public key as $(E_{pk'}(\mathbf{0}^l;\mathbf{r}_h), E_{pk'}(0;r_{h,l+1}))$. (Note that pk''s is chosen freshly for each query.)

Since $r_{h,l+1}$ for every h is sampled independently of $\mathbf{r}_h$, again we can easily see that any adversary that can distinguish between *Game*-3 and *Game*-4 can be reduced to break the $(l+1,n)$-RS security of $\mathcal{E}$ (which is a contradiction by Lemma (3)), implying *Game*-3 $\equiv^c$ *Game*-4. Finally, note that the entire view of an adversary under this game can be simulated by having only $(\mathbf{r}_1, r_{1,l+1},\ldots,\mathbf{r}_n, r_{n,l+1})$.

Game-5: This game runs exactly like *Game*-4, except that for $1 \le h \le n$ we now define $r_{h,l+1} = \mathbf{s}_h \cdot \mathbf{r}_h$. For every h by the Leftover Hash Lemma (in fact a special case of Lemma 2) we have that $(\mathbf{r}_h, \mathbf{s}_h \cdot \mathbf{r}_h)$ is statistically indistinguishable from a tuple chosen uniformly at random from R_n^{l+1}. Since n is poly-bounded, we obtain that *Game*-4 $\equiv^s$ *Game*-5. The view of the adversary under this game is exactly that under the n-projection security game when the challenge bit is one. This concludes the proof. □

2.5 Proof of Leakage Resilience

The following theorem shows the leakage resilience property of our scheme.

Theorem 3. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure public-key bit-encryption scheme providing PR-additive homomorphism and reproducibility. Then, the scheme built in Construction 1 is $(l - \log|\mathsf{R}_n| - u)$-length leakage resilient, for any $u \in \omega(\log n)$.*

Moreover, by taking $l = \omega(\log |\mathsf{R}_n| + u)$, the constructed scheme achieves a $(1 - o(1))$ resilience rate.

Proof. We first show the second statement of the theorem, assuming the first statement is true. Fix $u \in \omega(\log n)$. We know that the scheme provides $(l - \log |\mathsf{R}_n| - u)$-length leakage resilience, and so its resilience rate is

$$\frac{\omega\left(\log |\mathsf{R}_n| + u\right) - \log |\mathsf{R}_n| - u}{\omega(\log |\mathsf{R}_n| + u)} = 1 - \frac{\log |\mathsf{R}_n| + u}{\omega(\log |\mathsf{R}_n| + u)} = 1 - o(1). \tag{2.5.1}$$

To prove the first statement, first we assume without loss of generality that the adversary always outputs $(0, 1)$ as its challenge query, since otherwise the challenge ciphertext can be simulated by the adversary itself. We prove the first statement through a series of games, where the first game matches the actual leakage game (under a fixed challenge bit b), and in the last game the view of the adversary is independent of the challenge bit b. We conclude the proof by showing that the views of the adversary under any two adjacent games under the same $b \in \{0, 1\}$ are computationally indistinguishable. Thus, fix $b \in \{0, 1\}$ for the rest of the proof. In all game below, we let f be the leakage query of the adversary.

Game-0: In this game we reply to the adversary's queries exactly as in the actual leakage game, where the challenge bit is b. Thus, at the end of the game, the view of the adversary is $(c_1, \ldots, c_l, c_{l+1}, f(\mathbf{s}), c'_1, \ldots, c'_l, c'_{l+1})$, produced as follows: $\mathbf{s} \leftarrow \{0, 1\}^l$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}_n^l$, $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$, $(pk, *) \leftarrow G(1^n)$, $(pk', *) \leftarrow G(1^n)$, $c_i = E_{pk}(0; r_i)$, for $1 \le i \le l + 1$, $c'_j = E_{pk'}(0; r_j)$, for $1 \le j \le l$, and $c'_{l+1} = E_{pk'}(b; r_{l+1})$.

Notice that the view of the adversary may identically be produced as

$$(c_1, \ldots, c_l, c''_{l+1}, f(\mathbf{s}), c'_1, \ldots, c'_l, c'''_{l+1}), \tag{2.5.2}$$

where all c_i's and c'_i's are produced as above, and $c''_{l+1} = Hom(c_{h_1}, \ldots, c_{h_w})$ and $c'''_{l+1} = Hom(c'_{h_1}, \ldots, c'_{h_w}, E_{pk'}(b; 0))$ with $(h_1, \ldots, h_w)$ being the indices of non-zero bits of $\mathbf{s}$.

Game-1. In this game we generate the secret key, the public key and the response to the leakage query exactly as in *Game*-0, but we reply to the encryption challenge query in a special way. Formally, choose $\mathbf{s} \leftarrow \{0, 1\}^l$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}_n^l$, let $(h_1, \ldots, h_w)$ be the indices of non-zero bits of $\mathbf{s}$, and

- form the public key as $(c_1, \ldots, c_l, c''_{l+1})$, where $(pk, *) \leftarrow G(1^n)$, $c_i = E_{pk}(0; r_i)$, for $1 \leq i \leq l$, and $c''_{l+1} = Hom(c_{h_1}, \ldots, c_{h_w})$;
- reply to the the leakage query f with $f(\mathbf{s})$;
- return $(c'_1, \ldots, c'_l, c'''_{l+1})$ as the challenge ciphertext, where $\mathbf{b} = (b_1, \ldots, b_l) \leftarrow \{0,1\}^l$, $(pk', *) \leftarrow G(1^n)$, $c'_j = E_{pk'}(b_j; r_j)$, for $1 \leq j \leq l$, and

$$c'''_{l+1} = Hom(c'_{h_1}, \ldots, c'_{h_w}, E_{pk'}(b; 0)).$$

To show *Game*-0 $\equiv^G$ *Game*-1, note that both games can be simulated in exactly the same way by only having $\mathcal{D} = (c_1, \ldots, c_l, c'_1, \ldots, c'_l)$ (see Equation 2.5.2); this can be done by sampling $\mathbf{s}$ by ourselves and forming c''_{l+1} and c'''_{l+1} from, respectively, $(c_1, \ldots, c_l)$ and $(c'_1, \ldots, c'_l)$ by using the homomorphic property of $\mathcal{E}$. Further, since $\mathcal{E}$ is reproducible, in both games the distribution of $(c_1, \ldots, c_l)$ can be generated from $(c'_1, \ldots, c'_l)$ alone. Now since the distributions produced for $(c'_1, \ldots, c'_l)$ under the two games are computationally indistinguishable, which is followed by semantic security (recall that in *Game*-0, c'_i's are encryptions of zeros and in *Game*-1, they are encryptions of the bits of $\mathbf{b}$), we get that the distributions produced for $\mathcal{D}$ under the two games are computationally indistinguishable. Thus, we conclude *Game*-0 $\equiv^c$ *Game*-1. Notice that, under Game-1, the view of the adversary is

$$\big(E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l), E_{pk}(0; \mathbf{s} \cdot \mathbf{r}), f(\mathbf{s}), \\ E_{pk'}(b_1; r_1), \ldots, E_{pk'}(b_l; r_l), E_{pk'}(b_{l+1} + b; \mathbf{s} \cdot \mathbf{r})\big), \quad (2.5.3)$$

where $b_{l+1} = \mathbf{s} \cdot \mathbf{b}$.

<u>*Game*-2.</u> This game runs exactly as in *Game*-1 (Equation 2.5.3), except that now we generate $b_{l+1} \leftarrow \{0,1\}$, i.e., independent of $\mathbf{b} = (b_1, \ldots, b_l)$. First, notice that both *Game*-1 and *Game*-2 can be simulated in exactly the same manner by only having

$$Dis = (\mathbf{r}, \mathbf{s} \cdot \mathbf{r}, \mathbf{b}, b_{l+1}, f(\mathbf{s})). \quad (2.5.4)$$

The only difference between Dis from *Game*-1 to *Game*-2 is that under *Game*-1 we set $b_{l+1} = \mathbf{s} \cdot \mathbf{b}$, while in *Game*-2 we sample b_{l+1} freshly; the other parts of Dis are generated in the same way under both games: that is, $\mathbf{s} \leftarrow \{0,1\}^l$ and $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}_n^l$. Thus, to show the indistinguishability between these two

games, it suffices to show that the distributions of Dis under the two games are indistinguishable. We have,

$$\begin{aligned}\widetilde{H}_\infty(\mathbf{s}|\mathbf{r}, \mathbf{s}\cdot\mathbf{r}, f(\mathbf{s})) &\geq \widetilde{H}_\infty(\mathbf{s}|\mathbf{r}, f(\mathbf{s})) - log|\mathsf{R}_n| \\ &= \widetilde{H}_\infty(\mathbf{s}|f(\mathbf{s})) - log|\mathsf{R}_n| \\ &\geq H_\infty(\mathbf{s}) - l + \log|\mathsf{R}_n| + u - log|\mathsf{R}_n| \\ &= u = \omega(\log n).\end{aligned}$$

Now, since $\mathbf{r}$ is independent of $\mathbf{b}$, and also that f is independent of $\mathbf{b}$ (since f is queried before seeing the challenge ciphertext) we may use Lemma 2 to deduce that the distribution of Dis under $Game$-1 and $Game$-2 are statistically indistinguishable. To apply Lemma 2, take $D = \mathbf{s}$, $S = \mathbf{b}$ and $X = (\mathbf{r}, \mathbf{s}\cdot\mathbf{r}, f(\mathbf{s}))$. Notice that $Game$-2 produces the same views for the adversary under $b = 0$ and $b = 1$ (since b_{l+1} is chosen uniformly at random and hides the value of b), and hence the proof is complete. □

2.6 Proof of Auxiliary-Input Security

We now show that the encryption scheme produced in Construction 1 provides auxiliary-input security. We first consider weak-auxiliary-input security and then discuss the extension to the strong-auxiliary case.

Theorem 4. *Let $\mathcal{E} = (G, E, Dec, Hom, Rep)$ be a CPA-secure public-key bit-encryption scheme providing PR-additive homomorphism and reproducibility. Let $\mathcal{E}'$ be the scheme constructed from $\mathcal{E}$ using Construction 1. For any poly-bounded $l = l(n)$ and negligible function $\epsilon = \epsilon(n)$, it holds that $\mathcal{E}'$ is ϵ-weakly-auxiliary-input secure.*[1]

The proof of Theorem 4 follows similarly to that of Theorem 3, except for one step, where we replace real-randomness extraction with *pseudorandomness extraction*. We first give the following theorem, due to Goldreich and Levin [50], where we follow the presentation of [35], adapted to the binary field.

Theorem 5. *([50]) Assume that $l = l(n)$ and $h\colon \{0,1\}^l \to \{0,1\}^*$ is a (possibly randomized) function and $\mathcal{D}$ is a distinguisher, where*

$$|\Pr[D(\mathbf{b}, b, h(\mathbf{s})) = 1] - \Pr[D(\mathbf{b}, b', h(\mathbf{s})) = 1]| = \delta(l), \tag{2.6.1}$$

[1] In order for statement to be useful, it should hold that $\frac{1}{2^l} \leq \epsilon$, because otherwise the statement will be vacuously true, as $Aux^{st}_\epsilon = Aux^{wk}_\epsilon = \emptyset$.

where $\mathbf{s}, \mathbf{b} \leftarrow \{0,1\}^l$, $b \leftarrow \{0,1\}$ *and* $b' = \mathbf{s} \cdot \mathbf{b}$. *Then there exists an inverter* $\mathcal{A}$, *for which it holds that*

$$\Pr[\mathcal{A}(y) = \mathbf{s}] \in \Omega(\frac{\delta^3}{l}), \tag{2.6.2}$$

where $\mathbf{s} \leftarrow \{0,1\}^l$ *and* $y \leftarrow h(\mathbf{s})$.

We now give the proof of Theorem 4, using ideas from [35].

Proof. The proof follows by introducing *Game*-0, *Game*-1 and *Game*-2 exactly as in the proof of Theorem 3 (except that now the function f is applied to both the secret key and the public key), and deriving *Game*-0 $\equiv^G$ *Game*-1 exactly as in there. To prove *Game*-1 $\equiv^G$ *Game*-2, however, we proceed as below. To prove *Game*-1 $\equiv^G$ *Game*-2, it suffices to show that

$$\begin{aligned}(b_1, \ldots, b_l, b_{l+1}, f(PK, \mathbf{s}), \overbrace{E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l), E_{pk}(0; r_{l+1})}^{PK}) \equiv^c \\ (b_1, \ldots, b_l, b'_{l+1}, f(PK, \mathbf{s}), \underbrace{E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l), E_{pk}(0; r_{l+1})}_{PK}),\end{aligned} \tag{2.6.3}$$

where $\mathbf{s} \leftarrow \{0,1\}^l$, $b_1, \ldots, b_l, b_{l+1} \leftarrow \{0,1\}$, $b'_{l+1} = \mathbf{s} \cdot (b_1, \ldots, b_l)$, $\mathbf{r} = (r_1, \ldots, r_l) \leftarrow \mathsf{R}_n^l$, $r_{l+1} = \mathbf{s} \cdot \mathbf{r}$ and $(pk, *) \leftarrow G(1^n)$. The fact that proving Equation 2.6.3 suffices to conclude *Game*-1 $\equiv^c$ *Game*-2 can easily be verified by considering the descriptions of *Game*-1 and *Game*-2, taking into account the fact that the private-key scheme is reproducible.

By the assumption of the theorem, we know that it is ϵ-hard to recover $\mathbf{s}$ from $(PK, f(PK, \mathbf{s}))$. Now Equation 2.6.3 follows from Theorem 5, by defining the randomized function $h(\mathbf{s}) = (PK, f(PK, \mathbf{s}))$, where all the variables are sampled as above. Formally, if there is an adversary that can distinguish between the distributions in Equation 2.6.3 with a non-negligible probability, then there exists an adversary that, with a non-negligible probability, recovers $\mathbf{s}$ from $h(\mathbf{s}) = (PK, f(PK, \mathbf{s}))$, which is a contradiction to the first sentence of this paragraph. □

Remark 1. *As in previous work [35, 23] we can prove strong auxiliary-input security for* $\mathcal{E}'$ *with respect to subexponentially-hard functions by working with a modification of Construction 1, letting* $(c_1, \ldots, c_l) = (E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l))$ *be the public parameters of the scheme, and letting the public key be computed, under secret key* $\mathbf{s}$, *as* $Hom(c_{i_1}, \ldots, c_{i_w})$, *where* $(i_1, \ldots, i_w)$ *are the indices of non-zero bits of* $\mathbf{s}$. *Now since*

a public key under the new scheme has at most $l' = |\mathsf{R}_n|$ different values we can obtain $\frac{\epsilon}{l'}$-strong auxiliary-input security from ϵ-weak-auxiliary-input security. This last step follows since, for any scheme with l' different public keys, if recovering sk from $f(pk, sk)$ is $\frac{\epsilon}{l'}$-hard (i.e., succeeds with a probability at most $\frac{\epsilon}{l'}$), recovering sk from $(f(pk, sk), pk)$ is ϵ-hard. Finally, we mention that the proof of multiple-key circular security (Theorem 2) extends to the setting above which contains public parameters.

2.7 Open Problems

The main open problem is to construct circular-secure encryption from more general assumptions. The current homomorphic property we require forces a strong algebraic structure on the base scheme (by requiring the homomorphism to hold simultaneously with respect to both plaintexts and randomness), thereby making our assumptions specific in some sense. All existing circular-secure constructions heavily rely on similar homomorphic properties of their base assumptions. Giving a new construction without making any homomorphic assumption (or weaker than the kind we use) would be quite interesting and will probably need new techniques. We provide strong evidence in Chapter 4 that obtaining circular-secure encryption from CPA-secure encryption alone is most likely impossible.

Chapter 3

Applications of Reproducible Circular-Secure Encryption

In this chapter we show several applications of the notion of reproducible, circular-secure bit encryption. We first start with a general introduction, giving a description of the context of our results as well as an informal overview of the main techniques. We then formalize all the constructions and concepts sketched in Section 3.1 in subsequent sections.

3.1 Introduction

In this chapter we focus on another goal highlighted in Chapter 1: understanding what we can do with the notion of circular security. We give constructions of various fundamental cryptographic primitives based on a general bit-encryption scheme, which combines circular security with the reproducibility property (Definition 1). We show the following results:

1. We give a novel generic construction of trapdoor functions (TDFs) from reproducible bit encryption, and under this construction we show that successively stronger circular-security conditions result in successively stronger one-wayness conditions: we give a hierarchy of circular security notions, called *k-rec circular security*, all of which are weaker than those of [20, 23, 4], and show if the base scheme is k-rec circular secure, the constructed TDF is k-wise one-way, in the sense of [88].

2. We show how to extract many hardcore bits for our constructed TDFs, and by applying the results of [88] we obtain a black-box construction of CCA2-secure encryption from our assumptions. Our CCA2 construction is *non-shielding* in the sense of [47]. We partially justify this fact by showing with respect to a weaker encryption primitive than ours, a non-shielding black-box CCA2 construction is possible, while a shielding CCA2 construction is black-box impossible.

3. By slightly extending our base primitive, we show how to obtain deterministic encryption schemes secure under *block-source* inputs, as defined by [18].

4. We realize our base encryption primitive by showing the circular-secure schemes of [20, 23] are reproducible.

3.1.1 Background and Techniques

We assume the following notation and conventions throughout the introduction. Unless otherwise stated, an encryption scheme is bit encryption with randomness space (for encryption) $\{0,1\}^\rho$ and secret-key space $\{0,1\}^l$, where $l = l(n)$ and $\rho = \rho(n)$; by $E_{pk}(m)$, for $m \in \{0,1\}^*$, we mean bitwise encryption of m. Also, we use $E_{pk}(b;r)$ to denote encryption of bit b under randomness r.

Trapdoor functions. Central to public-key cryptography is the notion of *injective trapdoor one-way functions*, which refers to a family of functions, where each function in the family is easy to compute, but a randomly chosen function is hard to invert without a *trapdoor key*. A related notion is *witness-recovering CPA-secure encryption*: CPA-secure public-key encryption (PKE) where the decryption algorithm also recovers the randomness used for encryption. It is well-known that these two primitives are equivalent. However, as shown by Gertner *et al.* [48], there is a black-box separation between CPA-secure PKE and TDFs. An interpretation of this result is that a construction of a TDF from PKE should either be non-black-box, or should rely on specific properties of the PKE. Indeed, under specific assumptions, TDFs may be constructed "directly" (e.g., under the factoring assumption), or may be constructed by using the specifics of a particular PKE scheme (e.g., the strong homomorphisms, among other properties, of ElGamal encryption [83]).

A folklore attempt to build a TDF from PKE is to encrypt a message x under a randomness string derived deterministically from x. However, by [48], such a method-

ology is in general not sound. A naturally arising question is what properties of PKE enable sound realizations of this approach. The starting point of our work is a related question, namely: when does a PKE scheme allow "secure" encryption of r, using r itself as randomness? By security we mean it be hard to recover a random $r = r_1 \dots r_\rho \in \{0,1\}^\rho$ from

$$(E_{pk_1}(r_1; r), \dots, E_{pk_\rho}(r_\rho; r)),$$

where all pk_i's are chosen at random. Note that this immediately yields a TDF.

To address the above question we first introduce a simple manipulation technique that can be applied to a reproducible encryption scheme. Let $\mathcal{E} = (Gen, E, D, R)$ be a reproducible PKE scheme. Define $\mathcal{E}' = (Gen', E', D')$ as follows:

- Gen' samples (pk', sk'), where

  $$sk' = r \text{ and } pk' = c = E_{pk}(0; r)$$

  That is, the secret key is a (random) randomness string r and the public key is a dummy $\mathcal{E}$-ciphertext formed under randomness r;

- $E'_c(b)$ samples $(pk_1, sk_1) \leftarrow Gen$, computes

  $$c' = R(c, b, sk_1)$$

  and returns (pk_1, c') (i.e., E'_c encrypts b by reusing the randomness underlying c); and

- $D'_r(pk_1, c')$ returns the bit b for which $E_{pk_1}(b; r) = c'$.

Intuitively, CPA security of $\mathcal{E}'$ follows from reproducibility and CPA security of $\mathcal{E}$. Moreover, the construction swaps the key and randomness spaces of $\mathcal{E}$, and so the task of securely encrypting randomness in $\mathcal{E}'$ reduces to that of securely self-encrypting the secret key in $\mathcal{E}$; this latter problem is exactly the problem of circular security. The discussion above suggests a general technique for de-randomizing reproducible bit-encryption schemes, sketched below, which is the basis for all our subsequent constructions.

For $\mathcal{E} = (Gen, E, D, R)$ define a trapdoor function $\mathcal{F} = C(\mathcal{E}) = (G, F, F^{-1})$, where G, F and F^{-1}, are respectively the key-generation, evaluation and inversion

algorithms as follows. (See Section 3.2.1 for formal definitions and notation.) The domain space of F is the set of all pairs of public/secret keys generated under $Gen(1^n)$.

- G: To produce index/trapdoor keys (ik, tk), generate $(pk, sk) \leftarrow Gen(1^n)$, set

$$ik = (pk, E_{pk}(0; r_1), \ldots, E_{pk}(0; r_l)),$$

  for random r_i's, and set $tk = (r_1, \ldots, r_l)$.

- $F(\cdot, \cdot)$: On key $ik = (pk, c_1, \ldots, c_l)$ and domain input (pk', sk'), return $(pk', c'_1, \ldots, c'_l)$, where $c'_i = R(c_i, sk'_i, sk')$. (Here, sk'_i denotes the ith bit of sk'.)

- $F^{-1}(\cdot, \cdot)$: given trapdoor key $tk = (r_1, \ldots, r_l)$ and image point $(pk', c'_1, \ldots, c'_l)$, form the output as $(pk', b_1 \ldots b_l)$, where b_i is the bit satisfying $c'_i = E_{pk'}(b_i; r_i)$.

Correctness of $\mathcal{F}$ follows by the reproduction property of R. Also, since R is deterministic, so is the evaluation algorithm F. Finally, we take advantage of the fact that $\mathcal{E}$ is bit encryption to ensure efficient inversion for $\mathcal{F}$.

To discuss one-wayness we need the following definitions. For (pk, sk) output by Gen we refer to $E_{pk}(sk)$ as an *sk-self-encryption*. We call $\mathcal{E}$ *k-rec circular secure* if no adversary can recover (with a non-negligible chance) a random sk from k independent sk-self-encryptions, and call $\mathcal{E}$ *k-ind circular secure* if no adversary can distinguish between k independent sk-self-encryptions and encryptions of, say, zero. The notion of circular security in the literature is that of k-ind circular security, for unbounded k. For the construction above we show the following *tight* reduction.

Theorem 1. If $\mathcal{E}$ is reproducible and 1-rec circular secure then $C(\mathcal{E})$ is one-way.

The reduction above is "security preserving" in the following sense: assuming $\mathcal{E}$ is reproducible, then $\mathcal{E}$ is 1-rec circular secure iff $C(\mathcal{E})$ is one-way. Indeed, as we show next, by strengthening the condition of 1-rec circular security we achieve stronger forms of one-wayness.

A family of TDFs is called *k-wise one-way* [88] if one-wayness holds even if the given input is evaluated under k independently chosen functions.[1] More formally, we say that $\mathcal{F} = (G, F, F^{-1})$ is k-wise one-way, if $\mathcal{F}$'s *k-wise product*, defined as $F_{ik_1,\ldots,ik_k}(x) = (F_{ik_1}(x), \ldots, F_{ik_k}(x))$ is one-way. Rosen and Segev [88] showed the

[1] Actually, [88] chose another name for this particular notion, but we refer to it as k-wise one-wayness for simplicity.

utility of this notion by giving a black-box construction of CCA2-secure encryption based on k-wise one-way TDFs, for a sufficiently large k, generalizing a prior construction [83] based on lossy TDFs (LTDFs). Despite their utility, k-wise one-way TDFs (even for $k = 2$) are very strong primitives, whose only generic constructions have so far been based on LTDFs. Indeed, as shown by Vahlis [92], even 2-wise one-way TDFs cannot be constructed in a black-box way from trapdoor permutations (TDPs).

Our TDF construction provides an easy means of obtaining k-wise one-way TDFs: we can generalize Theorem 1 to show the following result.

If $\mathcal{E}$ is reproducible and k-rec circular secure then $C(\mathcal{E})$ is k-wise one-way.

To put our construction of k-wise one-way TDFs in context, we compare it to the LTDF-based construction [88]: the security reduction of [88] involves both statistical and computational arguments, allowing one to obtain only k-wise one-way TDFs for an *a priori* fixed but arbitrarily large value of k (which does suffice for CCA2 encryption) from sufficiently lossy TDFs. Our reduction argument, on the other hand, is entirely computational, allowing us to obtain unbounded k-wise one-way TDFs (i.e., a TDF that is k-wise one-way for any value of k) from the full circular security assumption.

As for the base assumptions, the relationships among the circular-security notions we described are not well-understood (beyond the trivial ones). Under certain assumptions these notions become equivalent. For example, any *re-randomizable* 1-rec circular-secure scheme is poly-ind circular secure: this follows by considering that a 1-rec circular-secure scheme is already poly-rec circular secure (because of re-randomizability), and that any poly-rec circular-secure scheme is also poly-ind circular secure [89, Theorem 8]. For the rest of the introduction, however, for ease of exposition, we describe the results with respect to full circular security.

We extend Construction C to the case in which the base scheme is t-circular secure (i.e., circular-secure with respect to t keys): the input of each TDF is t pairs of public/secret keys, the index key contains $l \cdot t$ dummy ciphertexts, and the evaluation algorithm on $(pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1})$ returns $(pk_0, \ldots, pk_{t-1})$ along with $t \cdot l$ ciphertexts formed by encrypting each bit of sk_i under $pk_{(i+1 \mod t)}$ (deterministically) by reusing the randomness of the corresponding ciphertext of the index key.

Extracting hardcore bits. Given the TDFs built above, we may apply the general Goldreich-Levin (GL) theorem [50] to extract a hardcore bit. We would like to,

however, avoid the use of the GL theorem for several reasons. First, the GL reduction, due to its generality, is not tight, while we would like to achieve CCA security with tight reductions. Second, for our deterministic encryption results we need to be able to extract many hardcore bits. Finally, since our base assumptions are strictly black-box-stronger (by Vahlis' result) than one-way TDFs, we should look for more specialized methods. We sketch below two deterministic methods for extracting many hardcore bits with tight security reductions for variants of our basic constructed TDFs. The first method applies to t-circular security and allows us to extract $\log((t-1)!)$ bits, with the advantage that it only increases the domain size of the basic TDF. The second method allows us to extract any, *a priori* fixed, number of bits, but it enlarges other spaces as well.

First method: a cycle hides its ordering. For simplicity, we describe the idea for 3-circular security, showing how to extract a single hardcore bit. The idea is that the notion of 3-circular security implies that no adversary can distinguish between the sequence $(pk_1, pk_2, pk_3, E_{pk_1}(sk_2), E_{pk_2}(sk_3), E_{pk_3}(sk_1))$ and the sequence $(pk_1, pk_2, pk_3, E_{pk_1}(sk_3), E_{pk_2}(sk_1), E_{pk_3}(sk_2))$. Now we augment our TDF construction described above (for the t-circular security case), so that the evaluation algorithm, besides the input $(pk_1, sk_1), (pk_2, sk_2), (pk_3, sk_3)$, also receives an additional bit b, used to dictate the ordering used to form the cycle. The inversion algorithm can open the ciphertexts, as before, and recover the bit b, by checking, say, whether the key encrypted under pk_1 is a secret key for pk_2 or for pk_3.[2] This technique extends to the t-circular security case for any $t > 3$, allowing us to "hide" a random ordering, providing $\log((t-1)!)$ hardcore bits.

Second method. We describe the idea for 1-circular security. We extend construction C above to be parameterized over an integer $m = m(n)$ and to result in a TDF whose input now consists of triples (pk, sk, x), as opposed to (pk, sk) alone, where $x \in \{0,1\}^m$. Moreover, we augment the index key to contain m added ciphertexts and let the trapdoor key contain their underlying randomness strings. Now $F(ik, (pk, sk, x))$ proceeds as before, but it also "encrypts" x in the process by again reusing randomness. For this TDF, we show that x remains pseudorandom even knowing $F(ik, (pk, sk, x))$. Finally, assuming the property that public keys under the base scheme are computed deterministically from their secret keys (plus perhaps some public parameters), we show how to obtain TDFs that hide a $(1 - o(1))$ fraction of

[2]This, however, imposes a negligible inversion error.

their input bits.

CCA-secure encryption. Using results on k-wise one-way TDFs with many hard-core bits,[3] we may now use the black-box construction of Rosen and Segev [88] to build a many-bit CCA2-secure PKE from a reproducible, circular secure bit-encryption scheme. Specifically, [88] gives a black-box construction of CCA2-secure encryption from k-wise one-way TDFs, for $k \in \Omega(n)$; they also show that $k \in \omega(\log n)$ suffices for CCA1 encryption. Our CCA constructions, by relying on that of [88], result in schemes whose decryption functions query the encryption function of the base scheme. Gertner, Malkin and Myers [47] refer to such constructions as *non-shielding*, and show that there exists no *shielding* black-box construction of CCA-secure from CPA-secure encryption. Since our base assumptions are black-box-stronger than CPA security, it is natural to ask whether the non-shielding nature of our CCA2 construction is just an artifact of the construction of [88], or whether it is inherent. We were not able to answer this question for our encryption primitive, mainly because of the presence of the reproduction function. However, we are able to answer this with respect to a weaker primitive than ours, which is a special case of *randomness-dependent-message-secure (RDMS)* encryption [13], which allows secure multiple bitwise-encryptions of a randomness string r under r itself as randomness (Formalized in Definition 8). Calling this new primitive RDMS encryption, we show that RDMS encryption is implied by our base assumptions, and also that it enables a non-shielding construction of CCA-secure encryption. We prove the latter by directly constructing k-wise one-way TDFs using RDMS encryption. Next we observe that the shielding black-box impossibility result of [47] extends even if the base scheme is an RDMS encryption primitive (Theorem 11). Indeed, it seems that this latter statement is true for most encryption primitives whose security requirements are defined with respect to passive indistinguishability (i.e., no decryption oracles); see Section 3.5.1 for more details. Thus, we obtain an encryption primitive, with respect to which a non-shielding black-box CCA-secure construction is possible, but under which a shielding CCA-secure construction is black-box impossible.

Deterministic encryption (DE). Following [18], a deterministic l-bit-encryption scheme is called (λ, l)-IND secure if encryptions of any two (efficient) λ-sources (i.e., distributions with min-entropy λ) result in computationally indistinguishable ciphertexts. We formulate two extended notions of circular security, called (λ, l)-*entropy*

[3] We note that our hardcore-security results hold not only for $\mathcal{F} = C(\mathcal{E})$, but also for $\mathcal{F}$'s k-wise products, under the corresponding assumptions. See Section 3.4.

circular security and *strong-(λ, l)-entropy circular security*, both of which require that circular security hold even if the secret key $sk \in \{0,1\}^l$ is sampled from a λ-source distribution, while the strong-entropy version requires one more assumption, related to the public-key distribution.[4]

We show our TDF construction immediately gives us a (λ, l)-IND-secure DE scheme if the base scheme satisfies strong (λ, l)-entropy circular security. We also show that, by appropriately choosing the parameters, the schemes of [20, 23] provide strong-entropy circular security, meaning that our generic transformation applies to these two schemes to obtain secure DE schemes, which explains the striking similarities between (especially) the DDH-based DE scheme of [18] and the scheme of Boneh *et al.* [20]. We also note that the extra condition of strong-entropy circular security may be satisfied if, informally, the key-generation algorithm acts as a *strong extractor*, producing the public key from the secret key (taken as the source) based on a public parameter (taken as the seed). Similar structural assumptions are made in other settings, e.g., [93], to obtain DE schemes.

For weak-entropy circular security we also show how to obtain a secure DE scheme, but with looser parameters, i.e., the (λ, l)-parameters of the base scheme are not maintained. We follow the so-called *encrypt-with-hardcore* technique, implicitly used in [12, 10, 18] and formalized in [42]. A high-level description of the idea is as follows. Assume $\mathcal{F} = (G, F, F^{-1})$ is a TDF with an associated hardcore function h producing $\Omega(n)$ hardcore bits, and we want to make $\mathcal{F}$ a secure DE scheme. Suppose we have the bonus that h preserves hardcore security even if x is sampled from a biased, high min-entropy distribution. Now we can build a DE scheme by encrypting the output of F using its own associated hardcore bitstring under a randomized encryption scheme $\mathcal{E}'$: that is, $E_{ik,pk}(x) = E'_{pk}(F(ik, x), h(x))$; decryption can be done using ik's trapdoor key and pk's secret key. Security of E comes from the fact that $(F(ik, x), h(x))$ is computationally indistinguishable from $(F(ik, x), r)$, so $h(x)$ is as good as a fresh randomness string. The only remaining issue is that E may require a longer randomness string, which, however, can be handled by applying a pseudorandom generator to $h(x)$.

[4]The notion of weak-entropy circular security was also considered by [24] in the context of KDM amplification.

3.1.2 Further Discussion

The possibility of obtaining lossy trapdoor functions. Since LTDFs [83] are the only generic assumption (to the best of our knowledge) that imply k-wise one-way TDFs, it is natural to ask about the relationship between LTDFs and our base primitive. We believe these notions are incomparable. First, under our encryption primitive, we are able to obtain a TDF that is k-wise one-way for unbounded k's; LTDFs are known to achieve bounded k-wise one-way TDFs, but this does not seem to generalize to the unbounded case, mainly due to the nature of LTDF-based proof techniques that also rely on statistical arguments. (See [88, Theorem 3.3].) On the other hand, LTDFs have powerful statistical properties (i.e., losing information in lossy mode) which do not seem to be realizable under our assumptions. In particular, we were not able to define "lossy" keys (in the sense of [83]) under our constructions; those lossy keys should be vectors of encryptions under the base scheme (as in injective keys) in such a way that when one applies the reproduction function to them (as in the evaluation algorithm) this results in loss of information. This idea does not seem to be implementable without making additional assumptions. The work of Hemenway and Ostrovsky [63] shows how to build LTDFs from a form of *lossy encryption*. It might be possible to obtain LTDFs by formulating and assuming an appropriate form of lossy encryption in our setting; we have not, however, investigated this direction.

Comparison with [32]. Choi and Wee [32], by abstracting the DDH-based TDF construction of Peikert and Waters [83], show how to obtain LTDFs from reproducible encryption that is homomorphic with respect to both messages and randomness. In what comes below we first compare our construction to that of Choi and Wee and then compare our underlying assumptions.

The construction of [32] results in (a) public keys that consist of $(\log|Rand| + \omega(\log n))^2$ base-ciphertexts (i.e., ciphertexts under the base scheme) and (b) ciphertexts that consist of $(\log|Rand| + \omega(\log n))$ base-ciphertexts. (Here $Rand$ is the randomness space of the base encryption function.) Assuming $\log|Rand| \in \Theta(n)$ this translates into quadratically large public keys and linearly large ciphertexts. Under our basic TDF construction both constructed public keys and ciphertexts consist of $\log|SK|$ base-ciphertexts, where SK is the secret-key space of the base scheme. For a concrete comparison, DDH-based instantiations of [83, 32, 41] give us schemes whose public keys and ciphertexts contain, respectively, $\Theta(n^2)$ and $\Theta(n)$ group elements. On the other hand, the DDH-based circular-secure scheme of Boneh *et al.* has ci-

phertexts with $\Theta(n)$ group elements and secret keys with $\Theta(n)$ bits. Thus, we obtain a DDH-based TDF with public keys and ciphertexts both consisting of $\Theta(n^2)$ group elements. (The size of ciphertexts can be cut down to $\Theta(n)$ by removing redundancies; see Construction 5.) Thus, we obtain no improvements in efficiency, despite the fact that our generic construction offers public keys and ciphertexts each containing a linear (in $\log |SK|$) number of base-ciphertexts. The same phenomenon also holds for concrete deterministic encryption schemes. However, our work shows that progress in improving the efficiency of BHHO might lead to improvements in efficiency of existing DDH-based TDFs or DE schemes.

Homomorphism versus circular security. The notions of homomorphism (in the sense of [32]) and circular security for an encryption scheme are qualitatively different as they concern structural (that is, syntactic) versus security properties. Interestingly though, all constructions of circular-secure schemes in the literature rely on certain homomorphic properties of their underlying algebraic assumptions [20, 23, 4]. However, it is not clear whether the existence of reproducible circular-secure encryption implies that of reproducible, homomorphic encryption. (If such an implication is proved then all our results will be subsumed by [32], since LTDFs imply all primitives we build in this chapter.) For one thing, a circular-secure scheme by itself does not necessarily provide the homomorphic property of [32] (or even weaker forms thereof). For example, under widely-believed assumptions one may construct a CCA2-secure, circular-secure scheme [28, 64], but homomorphic properties for such a scheme violate CCA2-security. Moreover, it seems hard to construct a homomorphic encryption scheme starting from a reproducible, circular-secure scheme that does not provide any homomorphic properties by itself.

Finally, as noted by Rosen and Segev [88], in light of their black-box impossibility result separating LTDFs from k-wise TDFs, there may be generic assumptions that yield k-wise TDFs, but not LTDFs; we believe that our encryption primitive is an example of those.

Shielding versus non-shielding constructions. We note that almost all black-box CCA2 constructions are non-shielding, e.g., [73, 88, 83], except for a few cases which rely on very powerful (and structurally different) primitives, e.g., [19].[5] Intuitively, the non-shielding property of those constructions is used to do consistency checks on ciphertexts, i.e., it allows a simulator, that typically does not have the

[5]The concepts of shielding/non-shielding only apply to encryption or TDF based constructions; see Definition 7.

entire decryption key, to ensure that a given ciphertext is indeed generated by the encryption algorithm. It would be interesting to explore if there exist weaker encryption primitives (than those we consider) for which the black-box separation of [47] is the best possible.

Non-bit encryption case. We informally discuss adaptations of Construction $C(\mathcal{E})$ to the case in which the secret-key space of $\mathcal{E}$ is a subset of its plaintext space Msg (which allows the secret key to be encrypted as a whole) and that reproducibility holds with respect to Msg. Let SK be the secret key space of $\mathcal{E}$. For this case we may substantially improve efficiency by having each index key contain only one $\mathcal{E}$-ciphertext, whose underlying randomness will be reused to self-encrypt the secret key $sk \in SK$ given as input to the evaluation algorithm. To perform inversion, however, we would need to rely on one more assumption: it is efficiently possible to recover m from pk, $E_{pk}(m;r)$ and r, for all pk, m and r. This last property by itself is satisfied by natural cryptosystems, e.g., ElGamal. Moreover, there is a standard (and straightforward) way to make any CPA-secure scheme (for which $SK \subseteq \mathcal{M}$) circular secure (again when the entire secret key is encrypted at once). This transformation, however, does not (necessarily) maintain this last, inversion-needed property. Thus, our results suggest that the CPA-to-one-shot-circular transformation may be non-trivial (and interesting) if it is to maintain this last mentioned property.

3.2 Preliminaries

We first start with some notation. If a randomized algorithm if $A(x_1, \ldots, x_m; r)$ outputs a tuple of strings, we let $A_i(x_1, \ldots, x_m)$ be the distribution formed by outputting the ith component of $A(x_1, \ldots, x_m)$. We denote by f^{-1} the inverse of an injective function f.

3.2.1 Trapdoor Functions and Various One-Wayness Conditions

In this section we review the standard notion of injective trapdoor functions, the notion of hardcore functions and various one-wayness conditions.

In the following definitions, let $D = \{D_n\}$ be an ensemble of sets, $\mathcal{D}_n$ be a distribution over D_n and $\mathcal{D} = \{\mathcal{D}_n\}$.

Definition 3. *(one-way injective trapdoor functions) A D-domain* collection of injective trapdoor functions *(TDFs)*[6] *is given by three algorithms $\mathcal{F} = (G, F, F^{-1})$ as follows. The probabilistic algorithm $G(1^n)$ randomly produces a pair (ik, tk) of* injective/trapdoor keys*; the deterministic algorithm $F(ik, \cdot)$ given $x \in D_n$ produces an* image *$y = F(ik, x)$; and $F^{-1}(tk, \cdot)$ given an image y returns a pre-image x. We require $\mathcal{F}$ satisfy the correctness condition stating*

$$\Pr\left[F^{-1}(tk, F(ik, x)) = x\right] = 1,$$

where the probability is taken over the choices of $(ik, tk) \leftarrow G(1^n)$ and $x \leftarrow D_n$. We stress that the input domain of $F(ik, \cdot)$ only depends on the security parameter 1^n. We use the notation $Domain(F)$ to refer to $D = \{D_n\}$.

We call $\mathcal{F}$ $\mathcal{D}$-one-way if if for any adversary $\mathcal{A}$,

$$\Pr\left[\mathcal{A}(ik, F(ik, x)) = x\right] = negl(n),$$

where the probability is taken over the choices of $(ik, tk) \leftarrow G(1^n)$, $x \leftarrow \mathcal{D}_n$ and $\mathcal{A}$'s coins.

Definition 4. *(k-wise TDF products and k-wise one-wayness [88]) The k-wise product of a D-domain TDF $\mathcal{F} = (G, F, F^{-1})$ is a D-domain TDF $\mathcal{F}^{(k)} = (G^{(k)}, F^{(k)}, F^{-1(k)})$ constructed as follows. The algorithm $G^{(k)}(1^n)$ first samples*

$$(ik_1, tk_1), \ldots, (ik_k, tk_k) \leftarrow G(1^n),$$

and lets $(ik_1, \ldots, ik_k)$ be the index key and $(tk_1, \ldots, tk_k)$ be the trapdoor key. On input $x \in D_n$, $F^{(k)}((ik_1, \ldots, ik_k), \cdot)$ returns $(F(ik_1, x), \ldots, F(ik_k, x))$. Finally, $F^{-1(k)}$ is defined as

$$F^{-1(k)}((tk_1, \ldots, tk_k), y) = F^{-1}(tk_1, y).$$

We say that $\mathcal{F}$ is k-wise $\mathcal{D}$-one-way if $\mathcal{F}^{(k)}$ is $\mathcal{D}$-one-way.

Note that 1-wise $\mathcal{D}$-one-wayness is the standard notion of $\mathcal{D}$-one-wayness defined in Definition 3.

Definition 5. *Let $\mathcal{F} = (G, F, F^{-1})$ be a D-domain TDF and $h = \{h_n\}$ be an ensemble of deterministic functions where $h_n : D_n \to \{0,1\}^{p(n)}$ (for some polynomial p). We*

[6] We use TDF to refer to a collection of injective trapdoor functions henceforth.

say that h is a $\mathcal{D}$-hardcore function *for $\mathcal{F}$ if for any adversary $\mathcal{A}$,*

$$\left|\Pr\left[\mathcal{A}(ik, F(ik,x), h(x)) = 1\right] - \Pr\left[\mathcal{A}(ik, F(ik,x), Unif_{\{0,1\}^{p(n)}}) = 1\right]\right| = negl(n),$$

where $(ik, tk) \leftarrow G(1^n)$ and $x \leftarrow \mathcal{D}_n$.

3.2.2 Definitions Related to Encryption Schemes

All encryption schemes that appear throughout this chapter, unless otherwise stated, are *bit-encryption* schemes. In our applications we need to work with a more general notion of encryption schemes (than those discussed in Chapter 2) involving *public parameters*, as formalized next.

A bit-encryption scheme $\mathcal{E} = (Param, Gen, E, Dec)$ is defined as follows. The *parameter-generation algorithm* $Param$ on input 1^n outputs a random parameter, par. The *key-generation algorithm* Gen on inputs 1^n and par generates a public/secret key $(pk, sk) \leftarrow Gen(1^n, par)$; we assume pk includes par, so we do not include par as an input to other algorithms. The *encryption algorithm* E on inputs 1^n, public key pk, bit b and randomness $r \in Rand_n$, outputs a ciphertext $c = E_{pk}(b; r)$. The *decryption algorithm* Dec takes a secret key sk and ciphertext c, and deterministically outputs a bit $b = Dec_{sk}(c)$. For correctness, we require

$$\Pr\left[Dec_{sk}(E_{pk}(b)) = b\right] = 1,$$

where $par \leftarrow Param(1^n)$, $(pk, sk) \leftarrow Gen(1^n, par)$ and $b \leftarrow \{0,1\}$. We will typically use $Rand = \{Rand_n\}$ to denote the underlying randomness space of the encryption algorithm of a scheme under consideration.

Assumption 1. *Throughout this chapter we make the following two assumptions about any encryption scheme $\mathcal{E} = (Param, Gen, E, Dec)$ under consideration.*

1. *For any n and any $par \in Param(1^n)$, all secret keys output by $Gen(1^n)$ are bitstrings of the same length. Thus, we have an associated secret-key-length function, usually denoted by l, which is a function of the security parameter.*

2. *In all security definitions that involve generating many public keys (e.g., multiple-key based security definitions) we assume all the underlying keys are sampled with respect to a fixed, random par sampled once and for all at the beginning of the underlying game.*

Given the assumptions above, henceforth we typically omit the inclusion of $Param$.

We now define different flavors of the notion of *circular security* [29, 20]. As notation, for $m \in \{0,1\}^*$ we extend E to define $E_{pk}(m) = (E_{pk}(m_1), \ldots, E_{pk}(m_{|m|}))$. Also, for $\mathbf{r} = (r_1, \ldots, r_t)$ and $m \in \{0,1\}^t$ we write

$$E_{pk}(m; \mathbf{r}) = (E_{pk}(m_1; r_1), \ldots, E_{pk}(m_t; r_t)).$$

Definition 6. *For an encryption scheme $\mathcal{E} = (Gen, E, Dec)$ we define the following notions.*

1. *We say $\mathcal{E} = (Gen, E, Dec)$ is k-rec t-circular secure if for every adversary $\mathcal{A}$,*

$$\Pr\left[\mathcal{A}(pk_1, \ldots, pk_t, \mathbf{c}_1, \ldots, \mathbf{c}_k) = sk_1\right] = negl(n),$$

where

$$(pk_1, sk_1), \ldots, (pk_t, sk_t) \leftarrow Gen(1^n)$$

and for every $1 \leq i \leq k$

$$\mathbf{c}_i \leftarrow \left(E_{pk_1}(sk_2), \ldots, E_{pk_{t-1}}(sk_t), E_{pk_t}(sk_1)\right).$$

2. *We say $\mathcal{E}$ is k-ind t-circular secure if $\mathcal{E}$ is CPA secure and also it holds that*

$$(pk_1, \ldots, pk_t, \mathbf{c}_1, \ldots, \mathbf{c}_k) \equiv^c (pk_1, \ldots, pk_t, \mathbf{c}'_1, \ldots, \mathbf{c}'_k),$$

where $\mathbf{c}_i$'s are generated as above and for $1 \leq i \leq k$,

$$\mathbf{c}'_i \leftarrow \left(E_{pk_1}(0^l), \ldots, E_{pk_t}(0^l)\right),$$

where $l = |sk_1|$. Note that we add CPA security as a separate condition because otherwise the definition may be satisfied trivially, e.g., consider an encryption scheme under which the secret key is always the all-zero string and the encryption function is the identity function. Also, we stress that in Cases 1 and 2 above all the underlying keys are generated with respect to a fixed, random parameter. (See Case 2 of Assumption 2.)

3.3 TDFs from Reproducible Bit Encryption

We begin by giving a construction that takes as input a reproducible bit-encryption scheme and produces a TDF. (The notion of reproducibility was introduced in Definition 1.) We then show how to achieve increasingly stronger guarantees of one-wayness for the constructed TDF from corresponding assumptions about the base encryption primitive. We tailor our construction to the t-circular security case (i.e., circular security with respect to t pairs of public/secret keys), meaning that we will obtain guarantees of one-wayness for the constructed TDF from t-circular security assumptions.

3.3.1 TDF Construction

Construction 2. *The following construction, that we call C_1, takes as input a reproducible bit-encryption scheme $\mathcal{E} = (Gen, E, Dec, R)$ and integer $t = t(n)$, and generates a TDF, $\mathcal{F} = (G, F, F^{-1})$, with domain space D^t, where $D = Sup(Gen(1^n))$. Let $l = l(n)$ be the length of a secret key output by $Gen(1^n)$, which is well-defined by Case 1 of Assumption 2. Also, we denote the randomness space of E by $Rand = \{Rand_n\}$.*

- *$G(1^n)$: Sample an injective/trapdoor key (ik, tk) as follows. Choose $(pk, sk) \leftarrow Gen(1^n)$, and for $1 \leq i \leq l$ and $0 \leq j \leq t-1$ choose $r_i^j \leftarrow Rand_n$. Now let*

$$tk = (r_1^0, \ldots, r_l^0, \ldots, r_1^{t-1}, \ldots, r_l^{t-1}), \text{ and}$$
$$ik = (pk, c_1^0, \ldots, c_l^0, \ldots, c_1^{t-1}, \ldots, c_l^{t-1}),$$

  where for $1 \leq i \leq l$ and $0 \leq j \leq t-1$, we set $c_i^j = E_{pk}(0; r_i^j)$.

- *F: On an injective key*

$$ik = (pk, c_1^0, \ldots, c_l^0, \ldots, c_1^{t-1}, \ldots, c_l^{t-1})$$

  and domain point

$$x = (pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1})$$

  return

$$F(ik, x) = (pk_0, \ldots, pk_{t-1}, c'^0_1, \ldots, c'^0_l, \ldots, c'^{t-1}_1, \ldots, c'^{t-1}_l),$$

where, denoting by b_i^j *the* i*th bit of* $sk_{j+1 \bmod t}$*, we define*

$$c'^j_i = R(c_i^j, b_i^j, sk_j). \tag{3.3.1}$$

- F^{-1}*: On a trapdoor key*

$$(r_1^0, \ldots, r_l^0, \ldots, r_1^{t-1}, \ldots, r_l^{t-1})$$

and image point

$$(pk_0, \ldots, pk_{t-1}, c'^0_1, \ldots, c'^0_l, \ldots, c'^{t-1}_1, \ldots, c'^{t-1}_l)$$

retrieve the corresponding pre-image

$$(pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1})$$

as follows: For $0 \leq j \leq t-1$ *recover* sk_j *bit-by-bit by encrypting back both* 0 *and* 1 *with the "corresponding" provided randomness (and under the corresponding public key) and finding the matching bit.*

Completeness of the constructed TDF follows by reproducibility. We point out a few remarks. First, the efficiency of the search performed by the inversion algorithm relies on the fact that each ciphertext is hiding a single bit, encrypted under randomness known to the inverter. Second, the construction is entirely black-box, also accessing (during evaluation) the reproduction function. Third, the construction extends to the non-bit-encryption case (i.e., when the base scheme is a reproducible scheme but not a bit-encryption scheme), by fixing a mapping from bits to two fixed plaintext messages and still continuing to encrypt the input secret key bit-by-bit using the reproduction function during the evaluation algorithm, but actually encrypting the plaintext message each bit is mapped to. For this case, the one-wayness of the constructed TDF reduces to bit-wise circular security of the base scheme (with respect to the fixed mapping).

3.3.2 Reducing k-wise One-Wayness to k-rec Circular Security

Theorem 6. *Assume $\mathcal{E}$ is a reproducible bit-encryption scheme and $\mathcal{F}$ is the TDF built from $\mathcal{E}$ in Construction 2 based on integer $t = t(n)$. Then, $\mathcal{E}$ is k-rec t-circular secure if and only if $\mathcal{F}$ is k-wise $\mathcal{D}$-one-way, where $\mathcal{D} = (Gen(1^n))^t$. Moreover, the reductions are tight.*

Proof. We give the proof for the case in which the base encryption scheme is k-rec 1-circular secure, i.e., with respect to a single pair of public/secret keys. The proof for the general case follows similarly. Thus, in the following we have $\mathcal{D} = Gen(1^n)$.

Recall that we use $l = l(n)$ to denote the length of a secret key outout by $Gen(1^n)$. Also, recall the following notation defined earlier. For $\mathbf{r} = (r_1, \ldots, r_l)$ and $m \in \{0,1\}^l$, we define $E_{pk}(m; \mathbf{r}) = (E_{pk}(m_1; r_1), \ldots, E_{pk}(m_l; r_l))$.

($\Rightarrow$) Assume that $\mathcal{E}$ is k-rec 1-circular secure and $\mathcal{A}$ is an adversary against the k-wise $\mathcal{D}$-one-wayness of $\mathcal{F}$, achieving advantage ϵ, namely

$$\epsilon(n) = \Pr\Big[\mathcal{A}(\underbrace{pk', E_{pk'}(sk'; \mathbf{r}_1), \ldots, E_{pk'}(sk'; \mathbf{r}_k)}_{\textbf{image}}, \underbrace{pk_1, E_{pk_1}(0^l; \mathbf{r}_1), \ldots, pk_k, E_{pk_k}(0^l; \mathbf{r}_k)}_{\textbf{ik}}) = (pk', sk')\Big]$$

where $(pk', sk'), (pk_1, sk_1), \ldots, (pk_k, sk_k) \leftarrow Gen(1^n)$ and $\mathbf{r}_1, \ldots, \mathbf{r}_k \leftarrow Rand_n^l$. Notice that **ik** contains concatenations of k independent injective keys under $\mathcal{F}$ and **image** contains concatenations of the images of a random domain input, (pk', sk'), under the k injective keys. We first note that if $\mathcal{A}$ were only given **image**, it could perfectly generate **ik** by itself, by sampling

$$(pk_1, sk_1), \ldots, (pk_k, sk_k) \leftarrow Gen(1^n)$$

and appropriately using the reproduction function R to build **ik**. Thus, $\mathcal{A}$'s ability to invert the TDF with advantage ϵ reduces to a new adversary, $\mathcal{B}$, recovering sk' from $(pk', E_{pk'}(sk'; \mathbf{r}_1), \ldots, E_{pk'}(sk'; \mathbf{r}_k))$ with the same advantage, ϵ.

($\Leftarrow$) This direction follows trivially. That is, any adversary that recovers sk' from

$$(pk', E_{pk'}(sk'; \mathbf{r}_1), \ldots, E_{pk'}(sk'; \mathbf{r}_k))$$

with probability γ can also trivially recover sk' from

$$\left(pk', E_{pk'}(sk'; \mathbf{r}_1), \ldots, E_{pk'}(sk'; \mathbf{r}_k), pk_1, E_{pk_1}(0^l; \mathbf{r}_1), \ldots, pk_k, E_{pk_k}(0^l; \mathbf{r}_k)\right),$$

with the same probability, by simply discarding the second half of the sequence. □

We conclude this subsection with the following observation about the structure of the TDF given in Construction 2.

Remark 2. *We call a D-domain TDF* certifiable *if membership in D is efficiently decidable. A drawback of Our TDF in Construction 2 (and all those that appear henceforth) is that the TDF is not in general certifiable, since for a given encryption scheme (Gen, E, Dec) checking whether $(pk, sk) \in Gen(1^n)$ is not necessarily efficiently decidable. (Doing the standard test of encrypting many bits under pk and decrypting them back under sk and looking for matches only gives us a necessary condition.) It remains open to determine whether under our assumptions a certifiable TDF can be built.*

3.4 Extracting Many Hardcore Bits

We present two deterministic methods for extracting many hardcore bits from variants of the TDF presented in Construction 2, with tight reductions to the indistinguishability variants of circular security assumptions. The first method applies to t-circular security for $t \geq 3$, allowing us to directly extract $\log((t-1)!)$ bits, by expanding only the domain space (of the TDF of Construction 2) by the same number of bits (but without affecting the sizes of the system's other parameters). The second method is less restrictive, allowing us to extract (from t-circular security, for any $t \geq 1$) $m(n)$ hardcore bits, where m is an arbitrary but *a priori* fixed poly function; this results in increasing the domain space by $m(n)$ bits and the image, index-key and trapdoor-key spaces by poly factors of $m(n)$. In particular, by choosing the parameter m appropriately we obtain TDFs that hide a $1 - o(1)$ fraction of their input bits.

3.4.1 First Hardcore Extraction Method

We begin with some notation. For integer $t > 0$ define $[t] = \{0, \ldots, t-1\}$. Also, for a set X we define

$$f(X) = \{f(x) \colon x \in X\}.$$

Let S contain all permutations f on $[t]$ where f induces only one *cycle*: $X \subseteq [t]$ is called a cycle under f if $X \neq \emptyset$ and $f(X) = X$. Note that $[t]$ is always a cycle under any permutation f over $[t]$, and thus a one-cycle permutation is one that has only the trivial cycle. Formally,

$$S = \left\{ f \colon [t] \to [t] \mid f \text{ is injective and } \ \forall X \subsetneq [t], X \text{ is not a cycle under } f \right\}.$$

Note that

$$|S| = (t-1)!.$$

Intuitively, each $f \in S$ defines a possible circular ordering of encrypting a sequence of t pairs of public/secret keys, by having pk_i encrypt $sk_{f(i)}$. The one-cycle property guarantees that we have a single, full cycle. For example, it is not the case that pk_1 encrypts sk_2, pk_2 encrypts sk_1 and the remaining keys encrypt each other in a circular manner. Fix

$$\mathcal{O} : [(t-1)!] \to S$$

to be an efficient index function defined using a canonical ordering of the elements of S. We assume the following notation about $\mathcal{O}$.

Notation 1. *We write $\mathcal{O}(i, x)$ to denote $f_i(x)$, where f_i is the ith permutation according to the ordering fixed on S. We also require that, for any $f \in S$, given*

$$sq = \{(0, f(0)), \ldots, (t-1, f(t-1))\},$$

it is possible to efficiently compute the index of f according to the ordering, which we (by slightly abusing the notation) denote by $\mathcal{O}^{-1}(sq)$.

We now proceed to describe the modified TDF construction and the associated hardcore function.

Construction 3. *Let $\mathcal{E} = (Gen, E, Dec, R)$, t and D^t be as in Construction 2. The domain space of the TDF, $\mathcal{F} = (G, F, F^{-1})$, we build is now $(D^t, [(t-1)!])$. Again, let $l = l(n)$ be the secret-key-length function of $\mathcal{E}$ and $Rand = \{Rand_n\}$ be the randomness space of E.*

- *$G(1^n)$: As in Construction 2.*

- *F: On an injective key*

$$ik = (pk, c_1^0, \ldots, c_l^0, \ldots, c_1^{t-1}, \ldots, c_l^{t-1})$$

and domain point

$$x = (pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1}, u)$$

do the following. First, set the indices

$$(ind_0, \ldots, ind_{t-1}) = (\mathcal{O}(u, 0), \ldots, \mathcal{O}(u, t-1)).$$

Informally, the output will be $pk_0, \ldots, pk_{t-1}$ together with a chain of encryptions, where pk_j encrypts the bits of sk_{ind_j}. Formally, return the tuple

$$(pk_0, \ldots, pk_{t-1}, c'^0_1, \ldots, c'^0_l, \ldots, c'^{t-1}_1, \ldots, c'^{t-1}_l),$$

where, for $0 \leq j \leq t-1$ and $1 \leq i \leq l$, denoting by b_i^j the ith bit of sk_{ind_j}, we set

$$c'^j_i = R(c_i^j, b_i^j, sk_j).$$

- *F^{-1} : On a trapdoor key*

$$(r_1^0, \ldots, r_l^0, \ldots, r_1^{t-1}, \ldots, r_l^{t-1})$$

and image point

$$(pk_0, \ldots, pk_{t-1}, c'^0_1, \ldots, c'^0_l, \ldots, c'^{t-1}_1, \ldots, c'^{t-1}_l)$$

do the following steps:

  - *recover $(x_0, \ldots, x_{t-1})$, where $x_j \in \{0,1\}^l$ for all j, as follows: to retrieve the ith bit of x_j for $1 \leq i \leq l$, encrypt both 0 and 1 under pk_j using randomness r_i^j and check the result against c'^j_i;*
  - *for each $0 \leq j \leq t-1$ let ind_j be the index for which it holds that pk_{ind_j} is the matching public key of x_j,[7] and let $sk_{ind_j} = x_j$. Form $sq =$*

[7]This can be done by encrypting many bits under the public key and decrypting them under a candidate secret key. This, however, results in a negligible inversion error.

$\{(0, ind_0), \ldots, (t-1, ind_{t-1})\}$; *return*

$$(pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1}, \mathcal{O}^{-1}(sq)).$$

Hardcore function: *For $\mathcal{F}$ given above we define $h\colon (D^t, [(t-1)!]) \to [(t-1)!]$ as*

$$h(pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1}, u) = u.$$

Correctness of the new TDF follows immediately. Note that Construction 2 is a special case of Construction 3, by forming the encrypted cycle with respect to the fixed permutation f defined as

$$f(j) = (j+1 \mod t).$$

In contrast, Construction 3 forms the encrypted cycle according to a permutation $f \in S$ provided as input to the evaluation algorithm, where, as we show below, a random choice of f is what is computationally hidden by the output of the evaluation algorithm.

As a main step toward proving that h is a hardcore function for $\mathcal{F}^{(k)}$ we show that the following two distributions are computationally indistinguishable:

$$\mathcal{D} = \Big(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(sk_{f(0)}), \ldots, E_{pk_{t-1}}(sk_{f(t-1)})}_{\mathbf{1st}} \\ , \ldots, \underbrace{E_{pk_0}(sk_{f(0)}), \ldots, E_{pk_{t-1}}(sk_{f(t-1)})}_{\mathbf{kth}}\Big),$$

$$\mathcal{D}' = \Big(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(0^l), \ldots, E_{pk_{t-1}}(0^l)}_{\mathbf{1st}}, \ldots, \underbrace{E_{pk_0}(0^l), \ldots, E_{pk_{t-1}}(0^l)}_{\mathbf{kth}}\Big),$$

where $f \leftarrow S$ and all (pk_i, sk_i) are random pairs of public/secret keys. Here we use $\mathbf{1st}, \ldots, \mathbf{kth}$ to denote copies of the underlying distribution. If $f\colon [t] \mapsto [t]$ is fixed to $f(i) = (i+1 \mod t)$ then $\mathcal{D} \equiv^c \mathcal{D}'$ is exactly the notion of k-ind t-circular security. In what follows we show a tight reduction from distinguishing between $\mathcal{D}$ and $\mathcal{D}'$, for a random $f \in S$, to breaking the notion of k-ind t-circular security. The reduction itself, a more generalized version of that described in Lemma 5, is relatively easy, but its proof of correctness is quite tedious. We first need to establish the following lemma.

Lemma 4. *Let Compose denote a transformation taking a function* $f\colon [t] \mapsto [t]$ *to another function* $g =^{def} Compose(f)\colon [t] \mapsto [t]$*, defined as*

$$g(i) = f^{(i)}(0),$$

where we define $f^{(0)}(n) = n$ *and*

$$f^{(i)}(n) = \underbrace{f(f(\dots f}_{i}(n)\dots)).$$

Letting $+$ *denote addition modulo* t*, we then have:*

1. *If* $f \in S$*, then* $g = Compose(f)$ *is one-to-one.*
2. *For distinct* $f_1, f_2 \in S$*, defining* $g_1 = Compose(f_1)$ *and* $g_2 = Compose(f_2)$*, we have* $g_1 \neq g_2$*.*
3. *Define a transformation* $Permute(\cdot)$ *that transforms* $f \in S$ *to* $h\colon [t] \mapsto [t]$ *as*

   $$h(i) = g^{-1}(g(i) + 1),$$

   where $g = Compose(f)$*. For any* $f \in S$ *we have* $Permute(f) \in S$*. Moreover, for any distinct* $f_1, f_2 \in S$ *we have* $Permute(f_1) \neq Permute(f_2)$*.*
4. *The two distributions* $Permute(Unif_S)$ *and* $Unif_S$ *are identically distributed.*

Proof. Note that Item 4 follows from Item 3, so we prove Items 1, 2 and 3.

Item 1 Suppose for some $f \in S$, $g = Compose(f)$ is not one-to-one, namely for some $0 \leq i < j \leq t-1$,

$$f^{(i)}(0) = f^{(j)}(0).$$

Define

$$X = \{f^{(i)}(0), f^{(i+1)}(0), \dots, f^{(j-1)}(0)\}.$$

Note that $|X| = j - i < t$ and so $X \subsetneq [t]$. However, it is easy to see that $f(X) = X$, which is a contradiction to the assumption that $f \in S$.

Item 2 Suppose $f_1, f_2 \in S$, $f_1 \neq f_2$ and $g_1 = Compose(f_1)$ and $g_2 = Compose(f_2)$. Suppose toward a contradiction that $g_1 = g_2$, namely

$$f_1^{(i)}(0) = f_2^{(i)}(0) \text{ for all } i \in [t]. \tag{3.4.1}$$

Since $f_1 \neq f_2$ for some $x \in [t]$ we have $f_1(x) \neq f_2(x)$. Since $g_1 \colon [t] \mapsto [t]$ is one-to-one (proved in the previous item), we have for some i that $g_1(i) = x$ or equivalently

$$f_1^{(i)}(0) = x.$$

Now by Equation 3.4.1 we have $f_2^{(i)}(0) = x$. Thus,

$$f_1(x) = f_1(f_1^{(i)}(0)) = f_1^{(i+1)}(0) = f_2^{(i+1)}(0) = f_2(f_2^{(i)}(0)) = f_2(x),$$

which is a contradiction to the earlier assumption that $f_1(x) \neq f_2(x)$.

Item 3 Let $f \in S$, $g = Compose(f)$ and h be defined as

$$h(i) = g^{-1}(g(i) + 1).$$

We first show $h \in S$. Suppose toward a contradiction that for $X = \{x_1, \ldots, x_m\}$ it holds that $h(X) = X$, where $m < t$. Again we recall that $+$ below denotes addition modulo t. We have

$$\begin{aligned}
\{x_1, \ldots, x_m\} &= h(\{x_1, \ldots, x_m\}) \\
&\Leftrightarrow \{x_1, \ldots, x_m\} = \{g^{-1}(g(x_1) + 1), \ldots, g^{-1}(g(x_m) + 1)\} \\
&\Leftrightarrow \{g(x_1), \ldots, g(x_m)\} = \{g(x_1) + 1, \ldots, g(x_m) + 1\} \stackrel{\text{def}}{=} X'.
\end{aligned}$$

Assume without loss of generality that $0 \leq g(x_1) < \cdots < g(x_m) \leq t - 1$. Since, $g(x_m) + 1 \in X'$ and $g(x_m)$ is the maximum element of X' we obtain $g(x_m) = t - 1$ and as a result $g(x_1) = 0$. Also, since $g(x_1) + 1 \in X'$ we have $1 \in X'$ and so $g(x_2) = 1$. Continuing using this argument we obtain $g(x_m) = m - 1 < t - 1$. However, this contradicts the previously established fact that $g(x_m) = t - 1$.

We now show that for any two distinct $f_1, f_2 \in S$, $Permute(f_1) \neq Permute(f_2)$. Let $g_1 = Compose(f_1)$, $g_2 = Compose(f_2)$, $h_1 = Permute(f_1)$ and $h_2 = Permute(f_2)$. From the statement we just proved we deduce $h_1, h_2 \in S$. Also, since $f_1 \neq f_2$ by Item 2 we have $g_1 \neq g_2$ and as a result

$$g_1^{-1} \neq g_2^{-1}. \tag{3.4.2}$$

Suppose to the contrary that $h_1 = h_2$. Thus,

$$h_1^{(i)}(0) = h_2^{(i)}(0), \text{ for all } i \in [t].$$

On the other hand, we claim

$$h_1^{(i)}(0) = g_1^{-1}(i)$$
$$h_2^{(i)}(0) = g_2^{-1}(i),$$

which imply

$$g_1^{-1}(i) = h_1^{(i)}(0) = h_2^{(i)}(0) = g_2^{-1}(i), \text{ for all } i \in [t]$$

which is a contradiction to Equation 3.4.2. We prove our claim for h_1 and the proof for h_2 is exactly the same. As the base case we have

$$h_1^{(0)}(0) \overset{\text{by definition}}{=} 0 = g_1^{-1}(0),$$

as desired. Now assume $h_1^{(i)}(0) = g_1^{-1}(i)$ for some $i < t-1$; we have

$$h_1^{(i+1)}(0) = h_1(h_1^{(i)}(0)) = h_1(g_1^{-1}(i)) = g_1^{-1}(g_1(g_1^{-1}(i)) + 1) = g_1^{-1}(i+1), \qquad (3.4.3)$$

as claimed. □ □

Lemma 5. *Let $\mathcal{E} = (Gen, E, Dec)$ be an arbitrary encryption scheme with the secret-key-length function $l = l(n)$, and let $t = t(n)$ and $k = k(n)$ be two arbitrary polynomials. Consider the following distributions:*

$$\mathbf{Dis}_1 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_1; \mathbf{r}_0^1), \ldots, E_{pk_{t-1}}(sk_0; \mathbf{r}_{t-1}^1), \ldots, E_{pk_0}(sk_1; \mathbf{r}_0^k), \ldots, E_{pk_{t-1}}(sk_0; \mathbf{r}_{t-1}^k)\Big)$$

$$\mathbf{Dis}_2 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)}; \mathbf{r}_0^1), \ldots, E_{pk_{t-1}}(sk_{f(t-1)}; \mathbf{r}_{t-1}^1), \ldots, E_{pk_0}(sk_{f(0)}; \mathbf{r}_0^k), \ldots, E_{pk_{t-1}}(sk_{f(t-1)}; \mathbf{r}_{t-1}^k), f\Big),$$

$$\mathbf{Dis}_3 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(0^l; \mathbf{r}_0^1), \ldots, E_{pk_{t-1}}(0^l; \mathbf{r}_{t-1}^1), \ldots, E_{pk_0}(0^l; \mathbf{r}_0^k), \ldots, E_{pk_{t-1}}(0^l; \mathbf{r}_{t-1}^k)\Big),$$

$$\mathbf{Dis}_4 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(0^l; \mathbf{r}_0^1), \ldots, E_{pk_{t-1}}(0^l; \mathbf{r}_{t-1}^1), \ldots, E_{pk_0}(0^l; \mathbf{r}_0^k), \ldots, E_{pk_{t-1}}(0^l; \mathbf{r}_{t-1}^k), f\Big),$$

where

$$\begin{aligned}(pk_0, sk_0), \ldots, (pk_{t-1}, sk_{t-1}) &\leftarrow Gen(1^n) \\ f &\leftarrow S \\ \mathbf{r}_0^1, \ldots, \mathbf{r}_{t-1}^1, \ldots, \mathbf{r}_0^k, \ldots, \mathbf{r}_{t-1}^k &\leftarrow Rand_n^l.\end{aligned} \tag{3.4.4}$$

There exists a randomized algorithm Convert satisfying the following two properties:

$$Convert(\mathbf{Dis}_1) \equiv \mathbf{Dis}_2$$
$$Convert(\mathbf{Dis}_3) \equiv \mathbf{Dis}_4$$

Moreover, if $\mathcal{E}$ is k-ind t-circular secure then

$$\mathbf{Dis}_2 \equiv^c \mathbf{Dis}_4,$$

and the reduction is tight.

Proof. Note that the "moreover" part follows from the existence of *Convert* with the stated properties, and thus in what follows we show how to construct *Convert* with the stated properties.

For an arbitrary

$$\mathbf{out} = (pk_0, \ldots, pk_{t-1}, \mathbf{c}_0^1, \ldots, \mathbf{c}_{t-1}^1, \ldots, \mathbf{c}_0^k, \ldots, \mathbf{c}_{t-1}^k)$$

we show how $Convert(\mathbf{out})$ works.

Below we use the operator $+$ to mean addition modulo t.

- Sample $f \leftarrow S$, and define $g = Compose(f)$ and $h = Permute(f)$ (recall that these two transformations were defined in Lemma 4.) That is,

$$\begin{aligned}g(i) &= f^{(i)}(0) \\ h(i) &= g^{-1}(g(i) + 1).\end{aligned}$$

- return

$$(pk_{g(0)}, \ldots, pk_{g(t-1)}, \mathbf{c}_{g(0)}^1, \ldots, \mathbf{c}_{g(t-1)}^1, \ldots, \mathbf{c}_{g(0)}^k, \ldots, \mathbf{c}_{g(t-1)}^k, h).$$

We now prove that *Convert* provides the desired properties. In the following let

$$\begin{aligned}(pk_0, sk_0), \ldots, (pk_{t-1}, sk_{t-1}) &\leftarrow Gen(1^n)\\ f &\leftarrow S\\ g(i) &= f^{(i)}(0)\\ h(i) &= g^{-1}(g(i)+1).\end{aligned}$$

We have that $Convert(\mathbf{Dis}_1)$ is identically distributed as

$$\begin{aligned}&Convert(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(sk_1), \ldots, E_{pk_{t-1}}(sk_0)}_{1st}, \ldots, \underbrace{E_{pk_0}(sk_1), \ldots, E_{pk_{t-1}}(sk_0)}_{kth})\\ &\equiv \Big(pk_{g(0)}, \ldots, pk_{g(t-1)}, \underbrace{E_{pk_{g(0)}}(sk_{g(0)+1}), \ldots, E_{pk_{g(t-1)}}(sk_{g(t-1)+1})}_{1st},\\ &\qquad \ldots, \underbrace{E_{pk_{g(0)}}(sk_{g(0)+1}), \ldots, E_{pk_{g(t-1)}}(sk_{g(t-1)+1})}_{kth}, h\Big)\end{aligned} \tag{3.4.5}$$

Now for $i \in [t]$ defining

$$(pk'_i, sk'_i) = (pk_{g(i)}, sk_{g(i)})$$

we may rewrite the distribution in Equation 3.4.5 as

$$\underbrace{(pk'_0, \ldots, pk'_{t-1}, \underbrace{E_{pk'_0}(sk'_{h(0)}), \ldots, E_{pk'_{t-1}}(sk'_{h(t-1)})}_{1st}, \ldots, \underbrace{E_{pk'_0}(sk'_{h(0)}), \ldots, E_{pk'_{t-1}}(sk'_{h(t-1)})}_{kth}, h)}_{\mathbf{out1}}. \tag{3.4.6}$$

In Equation 3.4.6 we used the fact that

$$sk'_{h(i)} = sk_{g(h(i))} = sk_{g(i)+1}.$$

Now since f is chosen uniformly at random from S, by Part 4 of Lemma 4 we have h is also uniformly distributed over S, and so **out1** is a random element according to distribution $\mathbf{Dis}_2$.

To show $Convert(\mathbf{Dis}_3) \equiv \mathbf{Dis}_4$ note that

$$Convert(\mathbf{Dis}_3) \equiv \Big(pk_{g(0)}, \ldots, pk_{g(t-1)}, \underbrace{E_{pk_{g(0)}}(0^l), \ldots, E_{pk_{g(t-1)}}(0^l)}_{1st}, \ldots, \underbrace{E_{pk_{g(0)}}(0^l), \ldots, E_{pk_{g(t-1)}}(0^l)}_{kth}, h \Big),$$

where all the variables are sampled as above. By Lemma 4 g is one-to-one and h is distributed uniformly over S, and so we obtain $Convert(\mathbf{Dis}_3) \equiv \mathbf{Dis}_4$, and the proof is complete. □

The following lemma is standard. We give a sketch of the proof for completeness.

Lemma 6. *Let $\mathcal{F} = (G, F, F^{-1})$ be a D-domain TDF and $h_n : D_n \to \{0,1\}^{p(n)}$ define an ensemble of deterministic functions (for some poly function p). Let $\mathcal{D}_n$ be a distribution over D_n, and let $\mathcal{D} = \{\mathcal{D}_n\}$. For any adversasry $\mathcal{A}$ achieving advantage $\epsilon = \epsilon(n)$ against the $\mathcal{D}$-one-wayness of $\mathcal{F}$, there exists an adversary $\mathcal{B}$ that*

$$\left| \Pr\left[\mathcal{B}(ik, F(ik,x), h(x)) = 1\right] - \Pr\left[\mathcal{B}(ik, F(ik,x), Unif_{\{0,1\}^{p(n)}}) = 1\right] \right| \geq \frac{\epsilon}{2},$$

where $(ik, tk) \leftarrow G(1^n)$, $x \leftarrow \mathcal{D}_n$ and $\mathcal{B}$'s random coins.

Proof. The adversary $\mathcal{B}(ik, y, u)$ works as follows: it runs $\mathcal{A}(ik, y)$ to receive x. If $F(ik, x) \neq y$ then $\mathcal{B}$ returns $b \leftarrow \{0,1\}$. If $F(ik, x) = y$, $\mathcal{B}$ returns 1 if $u = h(x)$, and returns 0 otherwise. The desired bound follows. □

We now prove the following theorem.

Theorem 7. *Let $\mathcal{F}$ and h be the TDF and hardcore function constructed according to Construction 3 based on $\mathcal{E} = (Gen, E, Dec, Rep)$ and $t = t(n)$. Assuming $\mathcal{E}$ is k-ind t-circular-secure, $\mathcal{F}$ is k-wise one-way and h is a hardcore function for $\mathcal{F}^{(k)}$.*

Proof. By Lemma 6 it suffices to show that h is a hardcore function for $\mathcal{F}^{(k)}$. Proving that h is a hardcore function for $\mathcal{F}^{(k)}$ boils down to showing that $\mathcal{D}'_1 \equiv^c \mathcal{D}'_2$, where

$$\mathcal{D}'_1 = \Bigg(\underbrace{pk'_1, E_{pk'_1}(0^l;\mathbf{r}^1_0), \ldots, E_{pk'_1}(0^l;\mathbf{r}^1_{t-1})}_{\mathbf{ik_1}}, \ldots, \underbrace{pk'_k, E_{pk'_k}(0^l;\mathbf{r}^k_0), \ldots, E_{pk'_k}(0^l;\mathbf{r}^k_{t-1})}_{\mathbf{ik_k}}$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)};\mathbf{r}^1_0), \ldots, E_{pk_{t-1}}(sk_{f(t-1)};\mathbf{r}^1_{t-1})}_{\mathbf{im_1}}, \ldots,$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)};\mathbf{r}^k_0), \ldots, E_{pk_{t-1}}(sk_{f(t-1)};\mathbf{r}^k_{t-1})}_{\mathbf{im_k}}, f \Bigg),$$

$$\mathcal{D}'_2 = \Bigg(\underbrace{E_{pk'_1}(0^l;\mathbf{r}^1_0), \ldots, E_{pk'_1}(0^l;\mathbf{r}^1_{t-1})}_{\mathbf{ik_1}}, \ldots, \underbrace{E_{pk'_k}(0^l;\mathbf{r}^k_0), \ldots, E_{pk'_k}(0^l;\mathbf{r}^k_{t-1})}_{\mathbf{ik_k}}$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)};\mathbf{r}^1_0), \ldots, E_{pk_{t-1}}(sk_{f(t-1)};\mathbf{r}^1_{t-1})}_{\mathbf{im_1}}, \ldots,$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)};\mathbf{r}^k_0), \ldots, E_{pk_{t-1}}(sk_{f(t-1)};\mathbf{r}^k_{t-1})}_{\mathbf{im_k}}, f' \Bigg),$$

where

$$\begin{aligned} (pk'_1, sk'_1), \ldots, (pk'_k, sk'_k), (pk_0, sk_0), \ldots, (pk_{t-1}, sk_{t-1}) &\leftarrow Gen(1^n) \\ f, f' &\leftarrow S \\ \mathbf{r}^1_0, \ldots, \mathbf{r}^1_{t-1}, \ldots, \mathbf{r}^k_0, \ldots, \mathbf{r}^k_{t-1} &\leftarrow Rand^l_n. \end{aligned} \tag{3.4.7}$$

Fix the above way of sampling variables in the following. Note that since $\mathcal{E}$ is reproducible, given only $(\mathbf{im_1}, \ldots, \mathbf{im_k})$ one can perfectly simulate the rest, namely $(\mathbf{ik_1}, \ldots, \mathbf{ik_k})$ is obtained by sampling $(pk'_1, sk'_1), \ldots, (pk'_k, sk'_k)$ and using the reproduction function. Thus, to prove $\mathcal{D}'_1 \equiv^c \mathcal{D}'_2$, it suffices to show

$$\mathcal{D}_1 \equiv^c \mathcal{D}_2, \tag{3.4.8}$$

where

$$\mathcal{D}_1 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)};\mathbf{r}^1_0), \ldots, E_{pk_{t-1}}(sk_{f(t-1)};\mathbf{r}^1_{t-1}), \ldots,$$
$$E_{pk_0}(sk_{f(0)};\mathbf{r}^k_0), \ldots, E_{pk_{t-1}}(sk_{f(t-1)};\mathbf{r}^k_{t-1}), f \Big),$$

$$\mathcal{D}_2 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_{f(0)}; \mathbf{r}_0^1), \ldots, E_{pk_{t-1}}(sk_{f(t-1)}; \mathbf{r}_{t-1}^1), \ldots, E_{pk_0}(sk_{f(0)}; \mathbf{r}_0^k), \ldots, E_{pk_{t-1}}(sk_{f(t-1)}; \mathbf{r}_{t-1}^k), f' \Big),$$

From Lemma 5 we have

$$\mathcal{D}_1 \equiv^c \mathcal{D}_3, \tag{3.4.9}$$

$$\mathcal{D}_3 = \Big(pk_0, \ldots, pk_{t-1}, E_{pk_0}(0^l; \mathbf{r}_0^1), \ldots, E_{pk_{t-1}}(0^l; \mathbf{r}_{t-1}^1), \ldots, E_{pk_0}(0^l; \mathbf{r}_0^k), \ldots, E_{pk_{t-1}}(0^l; \mathbf{r}_{t-1}^k), f \Big).$$

Applying Lemma 5 again we obtain $\mathcal{D}_2 \equiv^c \mathcal{D}_3$. Thus, we have

$$\mathcal{D}_1 \equiv^c \mathcal{D}_3 \equiv^c \mathcal{D}_2,$$

as desired. □

3.4.2 Second hardcore extraction method

The second construction allows us to extract any (*a priori* fixed) number of pseudo-random bits, where these bits are the last input block of the TDF.

Construction 4. *Let $\mathcal{E} = (Gen, E, Dec, R)$, t and D^t be as in Construction 2, and let $m = m(n)$ be an integer. The domain space of the TDF $\mathcal{F} = (G, F, F^{-1})$ we build is $(D^t, \{0,1\}^m)$. Let $l = l(n)$ be the secret-key-length function of $\mathcal{E}$ and $Rand = \{Rand_n\}$ be the randomness space of E.*

- *$G(1^n)$: Sample an injective/trapdoor key (ik, tk) as follows. Choose $(pk, sk) \leftarrow Gen(1^n)$, and for $1 \le i \le l$ and $0 \le j \le t-1$ choose $r_i^j \leftarrow Rand_n$. Also, for every $1 \le i \le m$ choose $r_i \leftarrow Rand_n$. Now let*

$$tk = (r_1^0, \ldots, r_l^0, \ldots, r_1^{t-1}, \ldots, r_l^{t-1}, r_1, \ldots, r_m), \text{ and}$$
$$ik = (pk, c_1^0, \ldots, c_l^0, \ldots, c_1^{t-1}, \ldots, c_l^{t-1}, c_1, \ldots, c_m),$$

where for $1 \le i \le l$ and $0 \le j \le t-1$, we set $c_i^j = E_{pk}(0; r_i^j)$, and for $1 \le i \le m$ we set $c_i = E_{pk}(0; r_i)$.

- *On an injective key*

$$ik = (pk, c_1^0, \ldots, c_l^0, \ldots, c_1^{t-1}, \ldots, c_l^{t-1}, c_1, \ldots, c_m)$$

and domain point

$$x = (pk_0, sk_0, \ldots, pk_{t-1}, sk_{t-1}, u)$$

return

$$F\,(ik, x) = (pk_0, \ldots, pk_{t-1}, c'^0_1, \ldots, c'^0_l, \ldots, c'^{t-1}_1, \ldots, c'^{t-1}_l, c'_1, \ldots, c'_m),$$

where, denoting by b_i^j the ith bit of $sk_{(j+1 \mod t)}$, we define

$$c'^j_i = R(c_i^j, b_i^j, sk_j). \tag{3.4.10}$$

Also, for $1 \leq h \leq m$ we define

$$c'_h = R(c_h, u_h, sk_0).$$

- *F^{-1}: as in prior constructions.*

Hardcore function: *For $\mathcal{F}$ given above, we let $h\colon (D^t, \{0,1\}^m) \to \{0,1\}^m$ be defined as*

$$h(pk_1, sk_1, \ldots, pk_t, sk_t, u) = u.$$

Correctness of inversion is again clear and we have security as follows.

Theorem 8. *Let $\mathcal{F}$ and h be the TDF and hardcore function constructed according to Construction 4 based on $\mathcal{E} = (Gen, E, Dec, Rep)$, $m = m(n)$ and $t = t(n)$. Assuming $\mathcal{E}$ is k-ind t-circular-secure, $\mathcal{F}$ is k-wise one-way and h is a hardcore function for $\mathcal{F}^{(k)}$.*

Proof. By Lemma 6 it suffices to show that h is a hardcore function for $\mathcal{F}^{(k)}$. To

prove this we need to show $\mathcal{D}'_1 \equiv^c \mathcal{D}'_2$, where

$$\mathcal{D}'_1 = \Bigg(\underbrace{pk'_1, E_{pk'_1}(0^l; \mathbf{r}^1_0), \ldots, E_{pk'_1}(0^l; \mathbf{r}^1_{t-1}), E_{pk'_1}(0^m; \mathbf{r}^1_t)}_{\mathbf{ik_1}}, \ldots,$$
$$\underbrace{pk'_k, E_{pk'_k}(0^l; \mathbf{r}^k_0), \ldots, E_{pk'_k}(0^l; \mathbf{r}^k_{t-1}), E_{pk'_k}(0^m; \mathbf{r}^k_t)}_{\mathbf{ik_k}},$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_1; \mathbf{r}^1_0), \ldots, E_{pk_{t-1}}(sk_0; \mathbf{r}^1_{t-1}), E_{pk_0}(u; \mathbf{r}^1_t)}_{\mathbf{im_1}}, \ldots,$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_1; \mathbf{r}^k_0), \ldots, E_{pk_{t-1}}(sk_0; \mathbf{r}^k_{t-1}), E_{pk_0}(u; \mathbf{r}^k_t)}_{\mathbf{im_k}}, u \Bigg),$$

$$\mathcal{D}'_2 = \Bigg(\underbrace{pk'_1, E_{pk'_1}(0^l; \mathbf{r}^1_0), \ldots, E_{pk'_1}(0^l; \mathbf{r}^1_{t-1}), E_{pk'_1}(0^m; \mathbf{r}^1_t)}_{\mathbf{ik_1}}, \ldots,$$
$$\underbrace{pk'_k, E_{pk'_k}(0^l; \mathbf{r}^k_0), \ldots, E_{pk'_k}(0^l; \mathbf{r}^k_{t-1}), E_{pk'_k}(0^m; \mathbf{r}^k_t)}_{\mathbf{ik_k}},$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_1; \mathbf{r}^1_0), \ldots, E_{pk_{t-1}}(sk_0; \mathbf{r}^1_{t-1}), E_{pk_0}(u; \mathbf{r}^1_t)}_{\mathbf{im_1}}, \ldots,$$
$$\underbrace{pk_0, \ldots, pk_{t-1}, E_{pk_0}(sk_1; \mathbf{r}^k_0), \ldots, E_{pk_{t-1}}(sk_0; \mathbf{r}^k_{t-1}), E_{pk_0}(u; \mathbf{r}^k_t)}_{\mathbf{im_k}}, u' \Bigg),$$

where

$$(pk'_1, sk'_1), \ldots, (pk'_k, sk'_k), (pk_0, sk_0), \ldots, (pk_{t-1}, sk_{t-1}) \leftarrow Gen(1^n) \tag{3.4.11}$$
$$u, u' \leftarrow \{0,1\}^m$$
$$\mathbf{r}^1_0, \ldots, \mathbf{r}^1_{t-1}, \ldots, \mathbf{r}^k_0, \ldots, \mathbf{r}^k_{t-1} \leftarrow Rand^l_n \text{ and } \mathbf{r}^1_t, \ldots, \mathbf{r}^k_t \leftarrow Rand^m_n.$$

Fix the above way of sampling variables in the following. Note that since $\mathcal{E}$ is reproducible, given only $(\mathbf{im_1}, \ldots, \mathbf{im_k})$ one can perfectly simulate the rest, namely $(\mathbf{ik_1}, \ldots, \mathbf{ik_k})$ is obtained by sampling $(pk'_1, sk'_1), \ldots, (pk'_k, sk'_k)$ and using the reproduction function. Thus, to prove $\mathcal{D}'_1 \equiv^c \mathcal{D}'_2$, it suffices to show $\mathcal{D}_1 \equiv^c \mathcal{D}_4$, where

$$\mathcal{D}_1 = \Big(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(sk_1), \ldots, E_{pk_{t-1}}(sk_0), E_{pk_0}(u)}_{1st}, \ldots, \underbrace{E_{pk_0}(sk_1), \ldots, E_{pk_{t-1}}(sk_0), E_{pk_0}(u)}_{kth}, u \Big),$$

$$\mathcal{D}_4 = \Big(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(sk_1), \ldots, E_{pk_{t-1}}(sk_0), E_{pk_0}(u)}_{1st}, \ldots, \underbrace{E_{pk_0}(sk_1), \ldots, E_{pk_{t-1}}(sk_0), E_{pk_0}(u)}_{kth}, u' \Big).$$

We now introduce $\mathcal{D}_2$ and $\mathcal{D}_3$ and show

$$\mathcal{D}_1 \equiv^c \mathcal{D}_2 \equiv^c \mathcal{D}_3 \equiv^c \mathcal{D}_4,$$

which will conclude the proof:

$$\mathcal{D}_2 = \Big(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(0^l), \ldots, E_{pk_{t-1}}(0^l), E_{pk_0}(u)}_{1st}, \ldots, \underbrace{E_{pk_0}(0^l), \ldots, E_{pk_{t-1}}(0^l), E_{pk_0}(u)}_{kth}, u \Big),$$

$$\mathcal{D}_3 = \Big(pk_0, \ldots, pk_{t-1}, \underbrace{E_{pk_0}(0^l), \ldots, E_{pk_{t-1}}(0^l), E_{pk_0}(u)}_{1st}, \ldots, \underbrace{E_{pk_0}(0^l), \ldots, E_{pk_{t-1}}(0^l), E_{pk_0}(u)}_{kth}, u' \Big).$$

Now, $\mathcal{D}_1 \equiv^c \mathcal{D}_2$ follows by k-ind t-circular security of $\mathcal{E}$; $\mathcal{D}_2 \equiv^c \mathcal{D}_3$ follows by CPA-security of $\mathcal{E}$; and $\mathcal{D}_3 \equiv^c \mathcal{D}_4$ follows by k-ind t-circular security of $\mathcal{E}$. □

Remark 3. *In many concrete settings, for a public-key encryption scheme* $\mathcal{E} = (Param, Gen, E, Dec)$ *with public parameters, we have that*

$$(par, pk, sk) \equiv (par, Pub(sk_u, par), sk_u),$$

where $par \leftarrow Param(1^n)$, $(pk, sk) \leftarrow G(1^n, par)$, $sk_u \leftarrow \{0,1\}^l$ *and* Pub *is a de-*

terministic function. That is, the secret key is chosen uniformly at random and the public key is obtained deterministically from the secret key and the public parameters. For such schemes we may easily modify Construction 4 so that, following the notation used in Construction 4, the index key ik is augmented with $par \leftarrow Param(1^n)$ *and that the evaluation function* F *no longer takes* pk_i*'s as input (so its entire input is a bitstring), by computing* $pk_i = Pub(sk_i, par)$ *on its own for* $0 \leq i \leq t-1$*. Now by taking* $m \in \omega(t \cdot l)$ *we obtain a TDF (from the assumptions stated in Theorem 8) that hides a* $(1 - o(1))$*-fraction of its input bits. Note that if one-wayness (as opposed to k-wise one-wayness) is desired we can have TDFs without public parameters as described above, i.e., by incorporating the parameter-generation algorithm of* $\mathcal{E}$ *into the key-generation algorithm of the constructed TDF. However, in order to have k-wise one-wayness the constructed TDF should also have a separate public-parameter generation algorithm.*

3.5 CCA-secure encryption

3.5.1 CCA-Secure Encryption from Reproducible Circular-Secure Bit Encryption

Rosen and Segev [88, Theorem 1] give a construction of a CCA1-secure encryption scheme from any $\omega(\log n)$-wise one-way TDF and of a CCA2-secure encryption from any $\Omega(n)$-wise one-way TDFs. Their constructions are fully-black-box and *non-shielding*, in the sense described below.

Recall the notion of *fully-black-box reductions* [86]; we use (C, R) to denote a fully-black-box reduction, where C denotes the construction and R denotes the reduction algorithm. In the definition below we review what it means for an encryption-based or TDF-based construction C to be *shielding*/*non-shielding* [47].

Definition 7. *An encryption-based black-box construction* $C = (G, E, D)$ *is called* shielding *if* D *never calls the encryption function of the oracle scheme. Formally, for any* $\mu = (g, e, d)$ *that implements an encryption scheme,* D^{μ} *never calls* e*. Similarly, a TDF-based construction* $C = (G, E, D)$ *is called* non-shielding *if* D *never calls the evaluation algorithm of the oracle scheme. A construction is called non-shielding if it is not shielding.*

We call (C, R) a shielding (resp., non-shielding) fully-black-box reduction if (1)

(C, R) is a fully-black-box reduction and (2) C is a shielding (resp., non-shielding) construction. We simply use the terms non-shielding/shielding black-box constructions to refer to non-shielding/shielding fully-black-box reductions.

The following result is from [88].

Theorem 9. *[88] There exists a non-shielding black-box construction of CCA1-secure (resp., CCA2-secure) encryption schemes from $\omega(\log n)$-wise one-way (resp., $\Omega(n)$-wise one-way) TDFs. In particular, the constructed decryption algorithm (for both CCA1 and CCA2 cases) D calls F and F^{-1}, the evaluation and inversion algorithms, of the base TDF.*

We may now use Theorem 9 and our results from the previous section to obtain CCA1 and CCA2 secure encryption schemes from our assumptions. Note that all TDF constructions we have presented have the property that the constructed inversion algorithm, F^{-1}, calls the encryption algorithm of the base reproducible encryption scheme. Thus, we have the following corollary.

Corollary 1. *There exists a non-shielding black-box construction of CCA1-secure (resp., CCA2-secure) encryption schemes from reproducible, $\omega(\log n)$-wise (resp., $\Omega(n)$-wise) t-circular-secure encryption schemes, for any t.*

3.5.2 Shielding versus Non-Shielding CCA-Secure Constructions

Gertner, Malkin and Myers [47] show that there are no shielding black-box construction of CCA1-secure encryption from CPA-secure encryption. In Corollary 1 we showed our assumptions (for appropriately chosen parameters) result in a non-shielding CCA1-secure encryption construction. Since our base assumptions are strictly stronger than CPA security (at least in a black-box sense), a natural question is whether or not it is possible to give a shielding construction based on our assumptions. We do not currently know the answer to this question, but as we show below, there exists an encryption primitive, which is implied by our assumptions, based on which a non-shielding black-box CCA1-construction is possible, but from which no shielding black-box CCA1-construction is possible.

Our new encryption primitive is an extension of CPA-secure encryption, requiring that security holds even when encrypting certain *randomness-dependent messages*.

The following definition is basically an adaptation of variants of those of [62, 13] to the bit-encryption case.

Definition 8. *A bit-encryption scheme $\mathcal{E} = (Gen, E, Dec)$ with randomness space $\{0,1\}^\rho$ is q-randomness-dependent-message (RDM) secure if*

$$\left(\left(E_{pk_1^1}(r_1;r),\ldots,E_{pk_\rho^1}(r_\rho;r)\right),\ldots,\left(E_{pk_1^q}(r_1;r),\ldots,E_{pk_\rho^q}(r_\rho;r)\right)\right)$$
$$\equiv^c \left(\left(E_{pk_1^1}(0;r),\ldots,E_{pk_\rho^1}(0;r)\right),\ldots,\left(E_{pk_1^q}(0;r),\ldots,E_{pk_\rho^q}(0;r)\right)\right),$$

where $r \leftarrow \{0,1\}^\rho$ and all public keys are chosen at random according to Gen. For better readability, we made the inclusion of the public keys implicit.

In the definition above, since we are encrypting the randomness string bit-by-bit, we should form each encryption under a fresh and independent public key. Otherwise, an adversary can easily distinguish between the two distributions. The reason is that an adversary given c_1 and c_2 for $c_1 = E_{pk}(b_1, r)$ and $c_2 = E_{pk}(b_2; r)$ can check whether $b_1 = b_2$.

q-RDM-secure encryption from our assumptions. We first show below that the notion defined above is implied by our assumptions. For simplicity, we show the implication from 1-circular security assumptions (i.e., circular security with respect to one pair of public/secret keys), although this generalizes to get implications from t-circular security assumptions.

Lemma 7. *Assume $\mathcal{E} = (Gen, E, Dec, R)$ is a reproducible, q-ind 1-circular-secure bit encryption scheme with public key space $\{0,1\}^{l_1(n)}$, secret-key space $\{0,1\}^{l_2(n)}$ and randomness space $Rand_n$, for any security parameter n. There exists a q-RDM secure encryption scheme based on $\mathcal{E}$.*

Proof. Let $l_1 = l_1(n)$, $l_2 = l_2(n)$ and $Rand = Rand_n$. Given $\mathcal{E} = (Gen, E, Dec, R)$ we define below a bit encryption scheme $\mathcal{E}' = (Gen', E', Dec')$ whose randomness space, $Rand'$, is the public/secret key space of $\mathcal{E}$, i.e., $Rand' = \{0,1\}^{l_1+l_2}$.

- $Gen'(1^n)$: sample $(pk, sk) \leftarrow Gen(1^n)$ and $r \leftarrow Rand$; form the public key as $pk_{const} = E_{pk}(0;r)$ and the secret key as $sk_{const} = r$.
- E': given public key $pk_{const} = c$, bit b and randomness (pk', sk') return

$$c_{const} = (pk', R(c, b, sk')).$$

- D': given secret key $sk_{const} = r$ and ciphertext (pk', c') return the bit b such that $E_{pk'}(b; r) = c'$.

To prove q-RDM security of $\mathcal{E}'$ it suffices to show (by reproducibility of $\mathcal{E}$) that

$$\begin{aligned}\left(pk', E_{pk'}(pk'; \mathbf{r_1^1}), E_{pk'}(sk'; \mathbf{r_2^1}), \ldots, E_{pk'}(pk'; \mathbf{r_1^q}), E_{pk'}(sk'; \mathbf{r_2^q})\right) \\ \equiv^c \left(pk', E_{pk'}(0^{l_1}; \mathbf{r_1^1}), E_{pk'}(0^{l_2}; \mathbf{r_2^1}), \ldots, E_{pk'}(0^{l_1}; \mathbf{r_1^q}), E_{pk'}(0^{l_2}; \mathbf{r_2^q})\right),\end{aligned}$$

where $(pk', sk') \leftarrow Gen(1^n)$, $\mathbf{r}_1^1, \ldots, \mathbf{r}_1^q \leftarrow \{0,1\}^{l_1}$ and $\mathbf{r}_2^1, \ldots, \mathbf{r}_2^q \leftarrow \{0,1\}^{l_2}$. The above indistinguishability follows easily from q-ind 1-circular security of $\mathcal{E}$. □

Non-shielding CCA1 construction from q-RDM-secure encryption. Next, we show q-RDM-secure encryption easily implies q-wise one-way TDFs, which we will use to show the existence of a non-shielding CCA1 construction. We sketch the construction based on a **q**-RDM-secure encryption scheme $\mathcal{E}$. Let $\mathcal{E}$'s randomness space be $\{0,1\}^\rho$, and define TDF $\mathcal{F} = (G, F, F^{-1})$ as follows. The algorithm $G(1^n)$ runs $Gen(1^n)$ ρ times to obtain $(pk_1, sk_1), \ldots, (pk_\rho, sk_\rho)$, and returns $ik = (pk_1, \ldots, pk_\rho)$ and $tk = (sk_1, \ldots, sk_\rho)$. The algorithm F has domain space $\{0,1\}^\rho$ and $F_{pk_1,\ldots,pk_\rho}(r)$ returns $(E_{pk_1}(r_1; r), \ldots, E_{pk_\rho}(r_\rho, r))$. The inversion algorithm F^{-1} works in the obvious way.

Now it is not hard to show if $\mathcal{E}$ is q-RDM secure, then $\mathcal{F}$ is q-wise one-way. Specifically, note that the view of an adversary against q-wise one-wayness of $\mathcal{F}$ is as

$$\left(E_{pk_1^1}(r_1; r), \ldots, E_{pk_\rho^1}(r_\rho, r), \ldots, E_{pk_1^q}(r_1; r), \ldots, E_{pk_\rho^q}(r_\rho, r)\right),$$

and also recall the definition of q-RDM security. (For better readability, we have removed the inclusion of pk_i^j's in the above equation.) Thus, by applying Theorem 9 we obtain the following.

Corollary 2. *For any $q \in \omega(\log n)$ there exists a non-shielding black-box construction of CCA1-secure encryption from q-RDM-secure bit-encryption.*

Impossibility of shielding CCA1-construction from q-RDM-secure encryption. We now show the black-box separation of [47], stating that there are no shielding black-box constructions of CCA1-secure encryption from CPA-secure encryption, extends even if the base scheme is q-RDM-secure, for any poly-bounded q. Combined with Corollary 2 this gives us an encryption primitive which permits a non-shielding

black-box CCA1-secure construction, but from which no shielding black-box CCA1-secure construction is possible. We first start with an informal description of the separation model of [47] and then give the formal definitions.

Specifically, [47] introduces a tuple of oracles $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w}, \mathbf{u})$, where $\mathcal{O}_1 = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ model an idealized encryption scheme (when the oracle is chosen at random), and $\mathcal{O}_2 = (\mathbf{d}, \mathbf{w})$ are two security-weakening components, which are defined based on $\mathcal{O}_1$. For any candidate construction $\mathcal{E} = (Gen^{\mathcal{O}_1}, Enc^{\mathcal{O}_1}, Dec^{\mathbf{g},\mathbf{d}})$ Gertner *et al.* prove that

1. there exists an oracle-adversary $\mathcal{A}^{\mathcal{O}}$, which is unbounded in time but poly-bounded in the number of oracle calls, that breaks the CCA1 security of $\mathcal{E}$ with very high probability, where the probability is taken over a random choice of $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w}, \mathbf{u})$ and all internal random coins of the CCA1 game (Formalized in Theorem 10 below);

2. no adversary $\mathcal{A}^{\mathcal{O}}$ that makes at most a polynomial number of queries can win against the CPA-security of $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ with better $1/2 + poly/2^n$ probability, where the probability is taken over the random choice of $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w}, \mathbf{u})$ and those of the adversary and the CPA-game [47, Theorem 1]. That is, a random $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ is CPA-secure in a very strong sense against any query-bounded oracle-adversary $\mathcal{A}^{\mathcal{O}}$.

Therefore, to rule out shielding fully-black-box constructions of CCA1-secure encryption from a new encryption primitive, it suffices to prove Item 2 above with respect to the new primitive. This is what we do below with respect to RDM secure encryption (Theorem 11). We first give the formal description of the oracles as used in [47].

Definition 9. *([47]) Define ψ, a distribution on oracles $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w}, \mathbf{u})$, defined for each $n \in \mathbb{N}$, as follows.*

- $\mathbf{g}\colon \{0,1\}^n \mapsto \{0,1\}^{3n}$ *is a random one-to-one function. Function* $\mathbf{g}$ *is considered as a key generator, with sk being the secret key and* $pk = \mathbf{g}(sk)$ *as the public key.*

- $\mathbf{e}\colon \{0,1\}^{3n} \times \{0,1\} \times \{0,1\}^n \mapsto \{0,1\}^{3n}$ *is a random one-to-one function.*

- $\mathbf{d}\colon \{0,1\}^n \times \{0,1\}^{3n} \mapsto \{0,1,\perp\}$ *is the unique function specified based on* $(\mathbf{g},\mathbf{e})$*, for which it holds* $\mathbf{d}(sk,c) = b$ *if there exists* $r \in \{0,1\}^n$ *s.t.* $\mathbf{e}(\mathbf{g}(sk),b,r) = c$*; otherwise,* $\mathbf{d}(sk,c) = \perp$*.*

- $\mathbf{w}\colon \{0,1\}^{3n} \to \{0,1\}^{3n\times n} \cup \{\perp\}$ *is a random function sampled as follows. For* $\mathbf{w}(pk)$*, if it holds that* $\mathbf{g}^{-1}(pk) = \emptyset$ *then* $\mathbf{w}(pk) = \perp$*; otherwise, sample the strings* $r_1,\ldots,r_n \leftarrow \{0,1\}^n$ *and return*

$$(\mathbf{e}(pk,sk_1,r_1),\ldots,\mathbf{e}(pk,sk_n,r_n)),$$

  where $sk = \mathbf{g}^{-1}(pk)$*.*

- $\mathbf{u}:\{0,1\}^{3n} \times \{0,1\}^{3n} \mapsto \{\top,\perp\}$ *is a deterministic function which returns* $\top$ *if there exists* sk*,* b *and* r *such that* $\mathbf{g}(sk) = pk$ *and* $\mathbf{e}(pk,b,r) = c$*, and returns* $\perp$*, otherwise.*

For consistency, we may sometimes write $\mathbf{e}(pk,b,r)$ *and* $\mathbf{d}(sk,c)$*, respectively, as* $\mathbf{e}_{pk}(b;r)$ *and* $\mathbf{d}_{sk}(c)$*.*

The following theorem, which is from [47], shows that, informally speaking, for any candidate shielding construction, there exists an inefficient adversary (which, however, makes a polynomial number of queries) that *almost always* breaks the CCA1 security of the constructed scheme.

Theorem 10. *([47]) Fix a shielding bit-encryption construction* (Gen,E,Dec)*. There exists a CCA1 adversary* $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2)$*, where* $\mathcal{A}$ *is poly-bounded in the number of queries but unbounded otherwise, for which it holds that*

$$\Pr\left[(m_0,m_1,\sigma) \leftarrow \mathcal{A}_1^{Dec^{\mathbf{g},\mathbf{d}}(SK),\mathcal{O}}(PK)\ ;\ \mathcal{A}_2^{\mathcal{O}}(c,\sigma) = b\right] \geq 1 - \frac{1}{n}, \tag{3.5.1}$$

where $\mathcal{O} = (\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u}) \leftarrow \psi$*,* $b \leftarrow \{0,1\}$*,* $(PK,SK) \leftarrow Gen^{\mathbf{g},\mathbf{e},\mathbf{d}}(1^n)$ *and* $c \leftarrow E^{\mathbf{g},\mathbf{e},\mathbf{d}}(m_b)$*. Note that* σ *is the state information that* A_1 *passes on to* A_2*. Also,* $Dec^{\mathbf{g},\mathbf{d}}(SK)$ *denotes the decryption oracle to which* $\mathcal{A}_1$ *has access. Note that since* A *is a CCA1 adversary only* A_1 *has access to* $Dec^{\mathbf{g},\mathbf{d}}(SK)$*.*

We give and prove the following theorem, a CPA version of which was proved in [47].

Theorem 11. *For any (possibly) inefficient adversary $\mathcal{A}$ that makes at most $p = p(n)$ queries (for some p), it holds that*

$$\Pr_{\mathcal{O}=(\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{w},\mathbf{u})\leftarrow\psi,\ \ b\leftarrow\{0,1\},\ ds_b\leftarrow\mathcal{DS}_b}\left[\mathcal{A}^{\mathcal{O}}(ds_b)=b\right] \leq \frac{1}{2}+\frac{poly(p)}{2^n} \tag{3.5.2}$$

where

$$\mathcal{DS}_0 \equiv \Big((pk_1^1, \mathbf{e}_{pk_1^1}(r_1;r)),\ldots,(pk_n^1,\mathbf{e}_{pk_n^1}(r_n;r)),$$
$$\ldots,(pk_1^q,\mathbf{e}_{pk_1^q}(r_1;r)),\ldots,(pk_n^q,\mathbf{e}_{pk_n^q}(r_n;r))\Big)$$
$$\mathcal{DS}_1 \equiv \Big((pk_1^1, \mathbf{e}_{pk_1^1}(0;r)),\ldots,(pk_n^1,\mathbf{e}_{pk_n^1}(0;r)),$$
$$\ldots,(pk_1^q,\mathbf{e}_{pk_1^q}(0;r)),\ldots,(pk_n^q,\mathbf{e}_{pk_n^q}(0;r))\Big),$$

in which $r \leftarrow \{0,1\}^n$ and pk_i^j, for $1 \le i \le n$ and $1 \le j \le q$, is formed by sampling $sk_i^j \leftarrow \{0,1\}^n$ and setting $pk_i^j = \mathbf{g}(sk_i^j)$.

Proof. We first fix some notation. Let Pub_{chal} be the set of public keys given to $\mathcal{A}$ as part of its input ds_b. (*chal* stands for challenge.) To be consistent with the above notation assume

$$Pub_{chal} = \{pk_1^1,\ldots,pk_n^1,\ldots,pk_1^q,\ldots,pk_n^q\}.$$

Let Pub be the set of public keys that $\mathcal{A}$ obtains by querying $\mathbf{g}$. Let $PubCiph_{chal}$ be the set of pairs of public keys/ciphertexts which $\mathcal{A}$ can retrieve as part of its input ds_b. Finally, let $PubCiph$ contains all elements of $PubCiph_{chal}$ plus those pairs of public keys/ciphertexts that $\mathcal{A}$ obtains by querying $\mathbf{e}$ and by querying $\mathbf{w}$.

First, we may assume that (1) $\mathcal{A}$ only calls its oracles on the security parameter n, (2) $\mathcal{A}$ never queries $\mathbf{u}$, (3) $\mathcal{A}$ only queries $\mathbf{w}$ on inputs $pk \in Pub_{chal}$ and (4) $\mathcal{A}$ never queries $\mathbf{d}$. We explain below why we can make these assumptions.

For (1), note that by Definition 9 the outputs of functions $(\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{u},\mathbf{u})$ on different security parameters are independent of each other. Thus, calling these functions on security parameters other than n gives no knowledge to the adversary as the adversary can sample those answers by itself.

For (2), for any $(pk,c) \notin PubCiph$, the query $\mathbf{u}(pk,c)$ is answered with $\perp$ except with an inverse-exponential probability (since $\mathbf{g}$ is length tripling and also $\mathbf{e}$ is "almost" length-tripling for fixed pk). If $(pk,c) \in PubCiph$, however, then $\mathcal{A}$ already

knows the answer and there is no point in calling $\mathbf{u}$.

For (3), first note that for any $pk \notin Pub_{chal} \cup Pub$, the query $\mathbf{w}(pk)$ is answered with $\perp$ except with an inverse-exponential probability (using the same reasoning as above). Also, if $pk \in Pub$, then $\mathcal{A}$ itself can sample the answer by querying $\mathbf{e}$, since $\mathcal{A}$ knows $\mathbf{g}^{-1}(pk)$.

Using similar reasoning, we can show that any query $\mathbf{d}(sk, c)$ that $\mathcal{A}$ trivially does not know the answer to is replied to with $\perp$ except with an inverse-exponential probability.

Now assuming $\mathcal{A}$ makes at most $p = poly(n)$ queries, by observation (3) we may assume that $\mathcal{A}$ never queries $\mathbf{w}$, but instead $\mathcal{A}$'s input includes $p \times n^2 \times q = poly(p)$ more ciphertexts, where for each public key $pk_i^j \in Pub_{chal}$, we include p bit-by-bit encryptions of $sk_i^j = \mathbf{g}^{-1}(pk_i^j)$ (so $p \times n$ encryptions for each public key and since we have nq public keys this gives us the above number).

Now since $\mathcal{A}$ only queries $\mathbf{g}$ and $\mathbf{e}$, we consider the following events.

- (a) $PubHit$: at least one of $\mathbf{g}$ queries results in some $pk \in Pub_{chal}$;
- (b) $CiphHit$: $\mathcal{A}$ makes a query $\mathbf{e}(pk, b, r)$, to get c, and it holds that $(pk, c) \in C_{chal}$.

If neither $PubHit$ nor $CiphHit$ holds, then we can show that the probability that $\mathcal{A}$ can determine b is at most $1/2 + poly(n)/2^n$. Moreover, both $PubHit$ and $CiphHit$ can easily be shown that occur with at most $poly(n)/2^n$ probability. □

Using standard techniques in black-box separations (especially applying the Borel-Cantelli lemma) Theorems 10 and 11 can be combined to obtain the following corollary.

Corollary 3. *For any poly-bounded q, there exists no shielding black-box construction of CCA1-secure encryption from q-RDM-secure encryption.*

We note that it seems that one can generalize Corollary 3 to rule out the existence of shielding black-box CCA1 constructions from a large class of encryption primitives whose security is defined in terms of indistinguishability against passive attacks (i.e., no decryption oracles). In other words, the black-box separation generalizes to any (base) security requirement that is "realized" by an ideal encryption scheme $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ in the presence of $(\mathbf{w}, \mathbf{u})$. For example, Corollary 3 still holds true if RDM security is replaced with circular security.

3.6 Constructions for Deterministic Encryption

We first start with some notation and definitions. Since a deterministic encryption scheme is syntactically the same as a TDF, we denote a deterministic-encryption scheme as $\mathcal{DE} = (G, F, F^{-1})$. For a function l we call $\mathcal{DE}$ an l-bit scheme if the plaintext space of $\mathcal{DE}$ on any security parameter n is $\{0,1\}^{l(n)}$. We start by giving a notion of security for deterministic encryption schemes, which is essentially the single-message, indistinguishability-based notion of [18]. See [18] for definitional equivalences.

Definition 10. *We say that a deterministic l-bit encryption scheme $\mathcal{DE} = (G, F, F^{-1})$ is secure with respect to indistinguishability of λ-source inputs (shortly, (λ, l)-IND secure) if for any λ-sources $\mathcal{M}_0$ and $\mathcal{M}_1$ over $\{0,1\}^l$, it holds that*

$$(ik, F_{ik}(\mathcal{M}_0)) \equiv^c (ik, F_{ik}(\mathcal{M}_1)),$$

where $(ik, tk) \leftarrow G(1^n)$.

3.6.1 Tools for Obtaining Deterministic Encryption

In definitions below, we explicitly include the parameter-generation algorithm, since the definitions delicately depend on the presence of public parameters. Throughout this section we work with randomized encryption schemes (as base schemes for obtaining DE schemes) whose key generation algorithms admit a special form stated in Remark 3 and reviewed below.

Definition 11. *We call a randomized encryption scheme $\mathcal{E} = (Param, Gen, E, Dec)$ with secret-key-length function $l = l(n)$ canonical if there exists a deterministic function Pub such that*

$$(par, pk, sk) \equiv (par, Pub(sk_u, par), sk_u),$$

where $par \leftarrow Param(1^n)$, $(pk, sk) \leftarrow G(1^n, par)$, $sk_u \leftarrow \{0,1\}^l$. That is, the secret key of a canonical scheme is chosen uniformly at random and the public key is obtained deterministically from the secret key and the public parameters. Henceforth, we will reserve Pub to denote the stated function of a canonical form scheme under consideration.

We start by defining an extended notion of circular security, requiring that circular security hold even if the secret key is sampled from a non-full-entropy distribution. For technical reasons, we need to allow some information about the secret key to be leaked, assuming the average min entropy of the secret key conditioned on the leaked information is high. The following definition generalizes a similar definition of [24] to the average case. We note it is possible to prove our results with respect to the weaker definition of [24], but the proofs become more complex.

Definition 12. *We say a canonical bit-encryption scheme $\mathcal{E} = (Param, Gen, E, Dec)$ with secret-key-length function l is (λ, l)-entropy circular secure if for any joint distribution $(\mathcal{SK}, \mathcal{X})$, where $\mathcal{SK}$ is a distribution over $\{0,1\}^l$, satisfying the condition $\tilde{H}_\infty(\mathcal{SK}|\mathcal{X}) \geq \lambda$, we have*

$$(par, pk, E_{pk}(sk), E_{pk}(1), x) \equiv^c (par, pk, E_{pk}(0^l), E_{pk}(0), x),$$

where $(sk, x) \leftarrow (\mathcal{SK}, \mathcal{X})$, $par \leftarrow Param(1^n)$ and $pk = Pub(sk, par)$. We stress that par is chosen independently of (sk, x).

We should clarify that Definition 12 is different from simply the combination of circular security and *leakage resilience* notions [1, 75]. Under the leakage-resilience model, the public/secret keys are chosen as spelled out by the scheme, but the leakage function f (to be evaluated on the secret key) is chosen by the adversary (after seeing the public key). Under our model, in contrast, the secret key may be chosen from a non-full-entropy distribution, but the leaked information (x above) is chosen independently of the random par.

Next we define another strengthening of the notion of [24], which adds the requirement that the public key distributions formed under high-entropy secret keys be computationally indistinguishable. This may be guaranteed if, e.g., Pub is a *strong randomness extractor* [78], as is the case with known circular-secure schemes [20, 23].

Definition 13. *Let $\mathcal{E} = (Param, Gen, E, Dec)$ be a canonical bit encryption scheme with secret-key-length function l. We say $\mathcal{E}$ is strongly-(λ, l)-entropy circular secure if*

- *(a) for any λ-source $\mathcal{SK}$ on $\{0,1\}^l$,*

$$(par, pk, E_{pk}(sk), E_{pk}(1)) \equiv^c (par, pk, E_{pk}(0^l), E_{pk}(0)),$$

where $sk \leftarrow \mathcal{SK}$, $par \leftarrow Param(1^n)$ and $pk = Pub(sk, par)$; and

- *(b) for any λ-sources $\mathcal{SK}_1$ and $\mathcal{SK}_2$ on $\{0,1\}^l$, it holds that*

$$(par, Pub(sk_1, par)) \equiv^c (par, Pub(sk_2, par)),$$

where $sk_1 \leftarrow \mathcal{SK}_1$, $sk_2 \leftarrow \mathcal{SK}_2$ and $par \leftarrow Param(1^n)$. Note that par is chosen independently from both sk_1 and sk_2.

As mentioned before, Condition (b) above in some sense states that Pub should act closely like a seeded randomness extractor.

3.6.2 Constructions

We first show that starting from a canonical reproducible bit encryption scheme which provides strong (λ, l)-entropy circular security, a slight variant Construction 2 immediately gives us a (λ, l)-IND secure deterministic scheme—i.e., it preserves the parameters.

Theorem 12. *Let $\mathcal{E} = (Param, Gen, E, Dec, R)$ be a canonical reproducible bit encryption scheme with secret-key-length function l and $\mathcal{DE} = C_1(\mathcal{E}, 1)$ be the DE scheme built in Construction 2 based on $\mathcal{E}$ and $t = 1$.*[8] *If $\mathcal{E}$ is strongly-(λ, l)-entropy circular secure $\mathcal{F}$ is (λ, l)-IND secure.*

Proof. Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two arbitrary λ-sources on $\{0,1\}^l$. We need to prove that $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$, where

$$\mathcal{DS}_1 \equiv (\underbrace{par, pk, E_{pk}(0^l; \mathbf{r})}_{\textbf{ik}}, \underbrace{pk_1, E_{pk_1}(sk_1; \mathbf{r})}_{\textbf{image}}), \text{ and}$$

$$\mathcal{DS}_2 \equiv (\underbrace{par, pk, E_{pk}(0^l; \mathbf{r})}_{\textbf{ik}}, \underbrace{pk_2, E_{pk_2}(sk_2; \mathbf{r})}_{\textbf{image}}),$$

are computationally indistinguishable, where $(pk, sk) \leftarrow Gen(1^n)$, $sk_1 \leftarrow \mathcal{D}_1$, $sk_2 \leftarrow \mathcal{D}_2$, $par \leftarrow Param(1^n)$, $pk_1 = Pub(sk_1, par)$, $pk_2 = Pub(sk_2, par)$ and $\mathbf{r} \leftarrow Rand_n^l$. Fix the described way of sampling variables in the following. From strong-(λ, l)-

[8] Here we are working with a modified version of Construction 2 stated in Remark 3. Note that the constructed deterministic encryption scheme does not have public parameters.

entropy circular security we obtain

$$\begin{aligned}(par, pk_1, E_{pk_1}(sk_1; \mathbf{r})) &\equiv^c (par, pk_1, E_{pk_1}(0^l; \mathbf{r})), \text{ and} \\ (par, pk_2, E_{pk_2}(sk_2; \mathbf{r})) &\equiv^c (par, pk_2, E_{pk_2}(0^l; \mathbf{r})).\end{aligned} \tag{3.6.1}$$

Now from Equation 3.6.1 and reproducibility of $\mathcal{E}$, we obtain

$$\begin{aligned}\mathcal{DS}_1 &\equiv^c (par, pk, E_{pk}(0^l; \mathbf{r}), pk_1, E_{pk_1}(0^l; \mathbf{r})), \text{ and} \\ \mathcal{DS}_2 &\equiv^c (par, pk, E_{pk}(0^l; \mathbf{r}), pk_2, E_{pk_2}(0^l; \mathbf{r})).\end{aligned} \tag{3.6.2}$$

Now since

$$(par, pk_1) \equiv^c (par, pk_2),$$

which is again implied by strong (λ, l)-entropy circular security of $\mathcal{E}$, we have

$$(par, pk, E_{pk}(0^l; \mathbf{r}), pk_1, E_{pk_1}(0^l; \mathbf{r})) \equiv^c (par, pk, E_{pk}(0^l; \mathbf{r}), pk_2, E_{pk_2}(0^l; \mathbf{r}))$$

and hence $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$, as desired. □

Next we show that the "weaker" entropy circular security assumption also gives rise to DE schemes, but with looser security bounds and under more inefficient constructions. Our construction employs the encrypt-with-hardcore (EWH) technique, described in the introduction.

As terminology, we say that a bit encryption scheme $\mathcal{E} = (Param, Gen, E, Dec, R)$ has a *bitstring ciphertext space* if there exists a polynomial p_c such that the ciphertext space of $\mathcal{E}$ is a subset of $\{0,1\}^{p_c}$: formally, for all n, all $par \in Param(1^n)$, all $(pk, sk) \in Gen(1^n, par)$ and all b it holds that all $Sup(E_{pk}(b)) \subseteq \{0,1\}^{p_c(n)}$. Similarly, we may define an encryption scheme with a bitstring ciphertext space or a TDF with a bitstring image space, etc.

Theorem 13. *Let $\mathcal{E} = (Param, Gen, E, Dec, R)$ be a canonical, reproducible, (λ, l)-entropy circular secure encryption scheme, with randomness space $Rand_n = \{0,1\}^{p_r}$, secret-key-length function $l = l(n)$ and with bitstring public-key and ciphertext spaces. There exists an $(l + p_r + u, 2l + p_r - \lambda)$-IND-secure deterministic encryption scheme, where $u \in \omega(\log n)$ is an arbitrary function.*

We give an outline of the proof first, and then proceed with the formal proof. The first step of the proof is to show that we can use reproducibility of $\mathcal{E}$ to encrypt

any arbitrarily-long bitstring, say of length $p = p(n)$, using a p_r-bit-long randomness string. (Recall that $Rand_n = \{0,1\}^{p_r}$.) This can be done (see Lemma 8) by defining a new PKE scheme whose public keys are vectors of p base-public-keys, $(pk_1, \ldots, pk_p)$, and in which the encryption function reuses randomness $r \in \{0,1\}^{p_r}$ to encrypt $m = m_1 \ldots m_p \in \{0,1\}^p$ as

$$(E_{pk_1}(m_1; r), \ldots, E_{pk_p}(m_p; r)).$$

Now consider the TDF given by Construction 4, based on $t = 1$ and $m = l + p_r - \lambda$. Define

$$hc(sk, x) = (hash, hash(x)),$$

where $hash\colon \{0,1\}^m \mapsto \{0,1\}^{p_r}$ is chosen from a *family of universal hash functions*. As the next step we show that hc is a hardcore function for the TDF. Having proved this, to be able to apply the EWH method, we need to show that $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$, for

$$\mathcal{DS}_1 \equiv (hash(x), hash, par, pk, E_{pk}(0^l; \mathbf{r}_1), E_{pk}(0^{|x|}; \mathbf{r}_1), E_{pk}(sk; \mathbf{r}_1), E_{pk}(x; \mathbf{r}_2)), \text{ and}$$
$$\mathcal{DS}_2 \equiv (y, hash, par, pk, E_{pk}(0^l; \mathbf{r}_1), E_{pk}(0^{|x|}; \mathbf{r}_1), E_{pk}(sk; \mathbf{r}_1), E_{pk}(x; \mathbf{r}_2)),$$

where $y \leftarrow \{0,1\}^{p_r}$, $par \leftarrow Param(1^n)$, $(sk, x) \leftarrow (\mathcal{SK}, \mathcal{X})$, $pk = Pub(sk, par)$ and $H_\infty(\mathcal{SK}, \mathcal{X}) \geq l + p_r + u$. (Also, $\mathbf{r}_1$ and $\mathbf{r}_2$ are chosen independently.) Now since by Lemma 1

$$\tilde{H}_\infty(\mathcal{SK}|\mathcal{X}) \geq \tilde{H}_\infty(\mathcal{SK}, \mathcal{X}) - \log|Sup(\mathcal{X})| \geq (l + p_r + u) - (l + p_r - \lambda) = \lambda + u$$

we may appeal to the (λ, l)-entropy circular security of $\mathcal{E}$ to replace $E_{pk}(sk; \mathbf{r}_1)$, in both $\mathcal{DS}_1$ and $\mathcal{DS}_2$, with an all-zero encryption. (That is, we deduce that, say, $\mathcal{DS}_1$ is indistinguishable from a distribution that is exactly the same as $\mathcal{DS}_1$ but in which $E_{pk}(sk; \mathbf{r}_1)$ is replaced with $E_{pk}(0^l; \mathbf{r}_1)$.) In the next step we do the same for $E_{pk}(x; \mathbf{r}_1)$ (i.e., we get rid of the occurrences of x as a plaintext). Finally, using the facts that

$$\tilde{H}_\infty(\mathcal{X}|\mathcal{SK}) \geq \tilde{H}_\infty(\mathcal{X}, \mathcal{SK}) - |Sup(\mathcal{SK})| \geq (l + p_r + u) - l = p_r + u,$$

and that $u \in \omega(\log n)$, we apply Lemma 2 to replace $hash(x)$ with a random string y.

We now give the formal proof of the theorem above. Before giving the proof, we need to establish some lemmas.

In the following we will introduce variants of the TDFs discussed earlier, with associated hardcore functions hc, where hc also takes as input the underlying index key ik, besides the domain input x to produce $hc(ik, x)$, i.e., hc also depends on the underlying index key.

We give the following simple lemma, showing that using reproducibility, one can obtain schemes with arbitrarily-large plaintexts using relatively short randomness.

Lemma 8. *Assuming the existence of a reproducible, CPA-secure bit-encryption scheme $\mathcal{E} = (Param, G, E, D)$ with randomness space $\{0,1\}^{p_r}$, for any poly function p there exists a CPA-secure p-bit encryption scheme $\mathcal{E}' = (Param', G', E', D')$ with the same randomness space $\{0,1\}^{p_r}$.*

Proof. Define $\mathcal{E}'$ as follows:

- $Param' = Param$;
- $G'(1^n; par)$: run $G(1^n; par)$ p times to produce $(pk_1, sk_1), \ldots, (pk_p, sk_p)$ and form the public key as $(pk_1, \ldots, pk_p)$ and the secret key as $(sk_1, \ldots, sk_p)$;
- E': on public key $pk_{ext} = (pk_1, \ldots, pk_p)$, message $m \in \{0,1\}^p$ and randomness $r \in \{0,1\}^{p_r}$ return
$$(E_{pk_1}(m_1; r), \ldots, E_{pk_p}(m_p; r)).$$
- D': clear.

Using a simple hybrid argument one can prove the CPA-security of $\mathcal{E}'$ based on the CPA-security and reproducibility of $\mathcal{E}$. To do this, for $1 \leq i \leq p+1$ define hybrid $\mathcal{D}_i$ under which an encryption of m under $(pk_1, \ldots, pk_p)$ and randomness r is produced as

$$\begin{aligned}
&(E_{pk_1}(0; r), \ldots, E_{pk_{i-1}}(0; r), E_{pk_i}(m_i; r), \ldots, E_{pk_p}(m_p; r)) \qquad 1 \leq i \leq p \\
&(E_{pk_1}(0; r), \ldots, E_{pk_{i-1}}(0; r), E_{pk_i}(0; r), \ldots, E_{pk_p}(0; r)) \qquad i = p+1.
\end{aligned}$$

Now note that $\mathcal{D}_1$ and $\mathcal{D}_{p+1}$ are identically distributed to, respectively, $E_{pk}(m)$ and $E_{pk}(0^p)$, and that $\mathcal{D}_1 \equiv^c \cdots \equiv^c \mathcal{D}_{p+1}$. □

We note that a similar version of Lemma 8 may be proved without assuming reproducibility, but by applying a pseudorandom generator (PRG) to stretch the

randomness for the encryption algorithm. However, since we need reproducibility for other purposes anyway, we work with the version of the lemma given above.

As terminology, we say that function f is k-bit-valued if f's output is always in $\{0,1\}^k$.

Theorem 14. *Let $\mathcal{E}_1$ be a reproducible, CPA-secure bit-encryption scheme with randomness space $\{0,1\}^{p_r}$. Let $\mathcal{F} = (G, F, F^{-1})$ be a TDF family with an associated p_r-bit-valued hardcore function hc, with a bitstring image space and suppose $Domain(F) = \{0,1\}^l$. Assume that for any λ-source distribution $\mathcal{M}$ on $\{0,1\}^l$ it holds that*

$$(ik, F_{ik}(x), hc(ik, x)) \equiv^c (ik, F_{ik}(x), s), \tag{3.6.3}$$

where $(ik, tk) \leftarrow G(1^n)$, $x \leftarrow \mathcal{M}$ and $s \leftarrow \{0,1\}^{p_r}$. Then, there exists a (λ, l)-IND-secure deterministic encryption scheme $\widetilde{\mathcal{DE}} = (\widetilde{G}, \widetilde{F}, \widetilde{F}^{-1})$.

Proof. Since $\mathcal{F}$ has a bitstring image space, let the image space of $\mathcal{F}$ be a subset of $\{0,1\}^{p_o}$. Given $\mathcal{E}_1$, by Lemma 8, we may assume the existence of a CPA-secure encryption scheme $\mathcal{E} = (Param, Gen, E, Dec)$, whose randomness space is $\{0,1\}^{p_r}$ and whose plaintext space is $\{0,1\}^{p_o}$. We define $\widetilde{\mathcal{DE}} = (\widetilde{G}, \widetilde{F}, \widetilde{F}^{-1})$ as follows.

- $\widetilde{G}(1^n)$: return (ik, pk) as the injective key and (tk, sk) as the trapdoor key, by sampling

$$par \leftarrow Param(1^n);\ (pk, sk) \leftarrow Gen(1^n, par) \text{ and } (ik, tk) \leftarrow G(1^n);$$

- $\widetilde{F}$: define

$$\hat{F}_{(ik,pk)}(m) = E_{pk}\left(F_{ik}(m); h_{ik}(m)\right);$$

- $\widetilde{F}^{-1}$: define

$$\hat{F}^{-1}_{(tk,sk)}(c) = F^{-1}_{tk}(Dec_{sk}(c)).$$

The completeness of $\widetilde{\mathcal{DE}}$ is clear. Now toward (λ, l)-IND-security of $\widetilde{\mathcal{DE}}$, we need to prove that for arbitrary λ-source distributions $\mathcal{M}_1$ and $\mathcal{M}_2$, it holds that $\mathcal{DS}_1 \equiv^c \mathcal{DS}_4$, where

$$\begin{aligned}\mathcal{DS}_1 &= (ik, pk, E_{pk}(F_{ik}(m); hc(ik, m)))\\ \mathcal{DS}_4 &= (ik, pk, E_{pk}(F_{ik}(m'); hc(ik, m')))\,.\end{aligned}$$

where $(ik, tk) \leftarrow G(1^n)$, $par \leftarrow Param(1^n)$, $(pk, sk) \leftarrow Gen(1^n, par)$, $m \leftarrow \mathcal{M}_1$ and $m' \leftarrow \mathcal{M}_2$. Fix this way of sampling variables in what comes below.

To prove $\mathcal{DS}_1 \equiv^c \mathcal{DS}_4$ we introduce $\mathcal{DS}_2$ and $\mathcal{DS}_3$ and show

$$\mathcal{DS}_1 \equiv^c \mathcal{DS}_2 \equiv^c \mathcal{DS}_3 \equiv^c \mathcal{DS}_4.$$

Define

$$\mathcal{DS}_2 = (ik, pk, E_{pk}(F_{ik}(m); s))$$
$$\mathcal{DS}_3 = (ik, pk, E_{pk}(F_{ik}(m'); s)),$$

where $s \leftarrow \{0,1\}^{p_r}$.

Recall that by the assumption on $\mathcal{F}$ we have

$$(ik, F_{ik}(m), hc(ik, m)) \equiv^c (ik, F_{ik}(m), s); \tag{3.6.4}$$
$$(ik, F_{ik}(m'), hc(ik, m')) \equiv^c (ik, F_{ik}(m'), s). \tag{3.6.5}$$

Now $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$ follows by Equation 3.6.4; $\mathcal{DS}_2 \equiv^c \mathcal{DS}_3$ follows by CPA security of $\mathcal{E}$; $\mathcal{DS}_3 \equiv^c \mathcal{DS}_4$ follows by Equation 3.6.4.

□

We are now ready to give the proof of Theorem 13.

Theorem 13. (restated) *Let $\mathcal{E} = (Param, Gen, E, Dec, R)$ be a canonical, reproducible, (λ, l)-entropy circular secure encryption scheme, with randomness space $Rand_n = \{0,1\}^{p_r}$, secret-key-length function $l = l(n)$ and with bitstring public-key and ciphertext spaces. There exists an $(l + p_r + u, 2l + p_r - \lambda)$-IND-secure deterministic encryption scheme, where $u \in \omega(\log n)$ is an arbitrary function.*

Proof. Fix $u \in \omega(\log n)$ and let $l_i = l + p_r + u$ and $l_o = 2l + p_r - \lambda$. Our goal is to build an (l_i, l_o)-IND-secure deterministic encryption scheme. To do so, since we already have $\mathcal{E}$, which is reproducible with randomness space $\{0,1\}^{p_r}$, by Theorem 14 it suffices to construct a TDF $\mathcal{F} = (G, F, F^{-1})$ with a bitstring image space and with an associated p_r-bit-valued hardcore function hc, which satisfies the following properties:

1. $Domain(F) = \{0,1\}^{l_o}$; and

2. For any l_i-source $\mathcal{M}$ over $\{0,1\}^{l_o}$,

$$(ik, F_{ik_{ext}}(x_{ext}), hc(ik_{ext}, x_{ext}) \equiv^c (ik_{ext}, F_{ik_{ext}}(x_{ext}), s),$$

where $x_{ext} \leftarrow \mathcal{M}$, $(ik_{ext}, tk_{ext}) \leftarrow G(1^n)$ and $s \leftarrow \{0,1\}^{p_r}$.

Thus, we focus on building $(\mathcal{F}, hc)$ with the properties above. To this end, we need a universal family $\mathcal{H}$ of hash functions from $\{0,1\}^{l+p_r-\lambda}$ to $\{0,1\}^{p_r}$.

We build $\mathcal{F} = (G, F, F^{-1})$ by instantiating Construction 4 with $\mathcal{E}$, integer $t = 1$ and integer $m = l + p_r - \lambda$, with the only difference that we augment the injective key with $hash \leftarrow \mathcal{H}$.[9] (Recall that for Construction 4 t denotes the number of public/secret key pairs and m is the number of bits added to the input of the TDF.)

Note that

$$Domain(F) = \{0,1\}^{l+l+p_r-\lambda} = \{0,1\}^{l_o}$$

so Property 1 above is satisfied. To define the associated hardcore function hc, for an injective key $ik_{ext} = (ik, hash)$ and domain point $x_{ext} = (sk, x) \in \{0,1\}^l \times \{0,1\}^{l+p_r-\lambda}$, we simply define

$$hc(ik_{ext}, x_{ext}) = hash(x).$$

For property 2 we need to show that for any arbitrary l_i-source $\mathcal{M}$, it holds that $\mathcal{DS} \equiv^c \mathcal{DS}'$, where

$$\mathcal{DS} \equiv (\underbrace{par, pk, E_{pk}(0^l; \mathbf{r}_1), E_{pk}(0^m; \mathbf{r}_2), hash}_{\mathbf{ik}}, \underbrace{pk', E_{pk'}(sk'; \mathbf{r}_1), E_{pk'}(x; \mathbf{r}_2)}_{\mathbf{image}}, \underbrace{hash(x)}_{\mathbf{hc}}), \text{ and} \tag{3.6.6}$$

$$\mathcal{DS}' \equiv (\underbrace{par, pk, E_{pk}(0^l; \mathbf{r}_1), E_{pk}(0^m; \mathbf{r}_2), hash}_{\mathbf{ik}}, \underbrace{pk', E_{pk'}(sk'; \mathbf{r}_1), E_{pk'}(x; \mathbf{r}_2)}_{\mathbf{image}}, \underbrace{s}_{\mathbf{hc}}),$$

in which $par \leftarrow Param(1^n)$, $(pk, sk) \leftarrow Gen(1^n, par)$, $\mathbf{r}_1 \leftarrow Rand_n^l$, $\mathbf{r}_2 \leftarrow Rand_n^m$, $hash \leftarrow \mathcal{H}$, $(sk', x) \leftarrow \mathcal{M}$, $pk' = Pub(sk', par)$ and $s \leftarrow \{0,1\}^{p_r}$. Since $\mathcal{E}$ is reproducible, for each of the above two distributions, given

$$(par, hash, \mathbf{image}, \mathbf{hc})$$

[9] We are again working with the modified version of Construction 2 stated in Remark 3.

one can perfectly simulate the rest. Thus, to show $\mathcal{DS} \equiv^c \mathcal{DS}'$ it suffices to prove $\mathcal{DS}_1 \equiv^c \mathcal{DS}_4$, where

$$\mathcal{DS}_1 \equiv (par, hash, \underbrace{pk', E_{pk'}(sk'; \mathbf{r}_1), E_{pk'}(x; \mathbf{r}_2)}_{\textbf{image}}, \underbrace{hash(x)}_{\textbf{hc}}), \text{ and} \tag{3.6.7}$$
$$\mathcal{DS}_4 \equiv (par, hash, \underbrace{pk', E_{pk'}(sk'; \mathbf{r}_1), E_{pk'}(x; \mathbf{r}_2)}_{\textbf{image}}, \underbrace{s}_{\textbf{hc}}),$$

We introduce two more distributions, $\mathcal{DS}_2$ and $\mathcal{DS}_3$, and will show that $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2 \equiv^c \mathcal{DS}_3 \equiv^c \mathcal{DS}_4$, which will conclude the proof.

Fix the above way of sampling variables. Define

$$\mathcal{DS}_2 \equiv (par, hash, pk', E_{pk'}(0^l; \mathbf{r}_1), E_{pk'}(0^m; \mathbf{r}_2), hash(x)), \tag{3.6.8}$$
$$\mathcal{DS}_3 \equiv (par, hash, pk', E_{pk'}(0^l; \mathbf{r}_1), E_{pk'}(0^m; \mathbf{r}_2), s).$$

Before proving the desired indistinguishability relations we give the following two facts, obtained from Lemma 1.

$$\tilde{H}_\infty(sk'|x) \geq H_\infty(sk', x) - (l + p_r - \lambda) = l_i - (l + p_r - \lambda) = \lambda + u \tag{3.6.9}$$
$$\tilde{H}_\infty(x|sk') \geq H_\infty(x, sk') - l = p_r + u. \tag{3.6.10}$$

We now proceed with the rest of the proof.
To prove $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$, note that by Equation 3.6.9 and the (λ, l)-entropy circular security (Definition 12) we have

$$(par, pk', E_{pk'}(sk'; \mathbf{r}_1), E_{pk'}(x; \mathbf{r}_2), x) \equiv^c \left(par, pk', E_{pk'}(0^l; \mathbf{r}_1), E_{pk'}(0^m; \mathbf{r}_2), x\right), \tag{3.6.11}$$

which imply $\mathcal{DS}_1 \equiv^c \mathcal{DS}_2$.

To prove $\mathcal{DS}_2 \equiv^c \mathcal{DS}_3$ it suffices to show

$$\mathcal{DS}'_2 = (hash, hash(x), sk') \text{ and } \mathcal{DS}'_3 = (hash, s, sk')$$

are (statistically) indistinguishable: this is because we can define a randomized algorithm A such that $A(\mathcal{DS}'_2) \equiv \mathcal{DS}_2$ and $A(\mathcal{DS}'_3) = \mathcal{DS}_3$: A samples par at random

and lets $pk' = Pub(sk', par)$ and also samples the rest of the variables appropriately. By Lemma 2 we have

$$\Delta\left(\mathcal{DS}'_2, \mathcal{DS}'_3\right) \leq \frac{1}{2}\sqrt{\frac{2^{p_r}}{2^{\tilde{H}_\infty(x|sk')}}} \leq \frac{1}{2}\sqrt{\frac{2^{p_r}}{2^{(p_r+u)}}} \leq \frac{1}{2^{u/2}} = negl(n), \tag{3.6.12}$$

where the second inequality follows from Equation 3.6.10.

To prove $\mathcal{DS}_3 \equiv^c \mathcal{DS}_4$ note that by Equation 3.6.11 we have

$$(par, pk', E_{pk'}(sk'; \mathbf{r}_1), E_{pk'}(x; \mathbf{r}_2)) \equiv^c \left(par, pk', E_{pk'}(0^l; \mathbf{r}_1), E_{pk'}(0^m; \mathbf{r}_2)\right), \tag{3.6.13}$$

which implies $\mathcal{DS}_3 \equiv^c \mathcal{DS}_4$. □

3.7 Realizations

In this section we show how to build, based on concrete assumptions, encryption schemes that provide reproducibility and also strong forms of circular security, i.e., (λ, l)-strong circular security for an appropriate setting of parameters.

Throughout this section we will be working with multiplicative notation for groups. For a group element g we denote the inverse of g by g^{-1} and define $g_1/g_2 = g_1 \cdot g_2^{-1}$. We also denote the identity element by 1, and we define $g^0 = 1$, and for integer $x > 1$, $g^x = g \cdot g^{x-1}$. For an integer x we define $g^{-x} = (g^x)^{-1}$. If $\mathbf{g} = (g_1, \ldots, g_l)$ and r is an integer we define $\mathbf{g}^r = (g_1^r, \ldots, g_l^r)$. Finally, we define $(b_1, \ldots, b_l) \odot (g_1, \ldots, g_l) = \prod_{1 \leq i \leq l} g_i^{b_i}$.

3.7.1 From the Decisional Diffie-Hellman (DDH) Assumption

Let $\mathcal{G}$ be a *group scheme*, that is, a PPT algorithm that on input 1^n, outputs $(\mathbb{G}, g, o)$, where $\mathbb{G}$ is the description of a group, $g \in \mathbb{G}$ and $o = |\mathbb{G}|$ is a prime number. We say that $\mathcal{G}$ is *DDH hard* if

$$\left\{\mathbb{G}, |\mathbb{G}|, g_1, g_2, g_1^d, g_2^d\right\}_{n\in\mathbb{N}} \equiv^c \left\{\mathbb{G}, |\mathbb{G}|, g_1, g_2, g_3, g_4\right\}_{n\in\mathbb{N}},$$

where $\mathbb{G}$ is chosen by running $\mathcal{G}(1^n)$, $g_1, \ldots, g_4 \leftarrow \mathbb{G}$ and $d \leftarrow \mathbb{Z}_{|\mathbb{G}|}$.

We present the encryption scheme of [20], which we refer to as the BHHO scheme, below.

Definition 14. *(From [20]) Define $\mathcal{E} = (Param, Gen, E, Dec)$, which is parameterized over an integer $l = l(n)$ (which we instantiate later), as follows.*

- *$Param(1^n)$: Produce $(\mathbb{G}, g, o) \leftarrow \mathcal{G}(1^n)$ and return $par = (\mathbb{G}, g, \mathbf{g})$, where $\mathbf{g} \leftarrow \mathbb{G}^l$;*
- *$Gen(1^n)$: Sample the secret key as $sk \leftarrow \{0,1\}^l$ and set the public key $pk = sk \odot \mathbf{g}$;*
- *$E_{pk}(g_1; r))$: Sample $r \leftarrow \mathbb{Z}_q$ and return $(\mathbf{g}^r, pk^r \cdot g_1)$; and*
- *$D_{sk}((\mathbf{g}', g'))$: Clear from the encryption algorithm.*

Reproducibility. We now verify the reproducibility property with respect to every fixed choice of *par*. To do this, we need to show that from

$$\left(\underbrace{\mathbb{G}, g, \mathbf{g}}_{par}, \underbrace{sk_1 \odot \mathbf{g}}_{pk_1}, \underbrace{(\mathbf{g}^r, pk_1^r \cdot g_1)}_{E_{pk_1}(g_1;r)}, \underbrace{g_2}_{\text{target message}}, \underbrace{sk_2}_{\text{target secret key}} \right),$$

one can compute $(\mathbf{g}^r, (sk_2 \odot \mathbf{g})^r \cdot g_2)$; this is easy to see considering that the last quantity is indeed $(\mathbf{g}^r, (sk_2 \odot \mathbf{g}^r) \cdot g_2)$, and that all of $\mathbf{g}^r$, sk_2 and g_2 are provided in the input tuple.

We show below the optimized version of the instantiation of our general TDF construction using the BHHO scheme. By optimized we mean we have removed all redundancies created under the "raw" instantiation.

Construction 5. *The TDF is parameterized over $l = l(n)$. See Theorem 15 on how to instantiate l.*

- *G: sample $(\mathbb{G}, g, o) \leftarrow \mathcal{G}(1^n)$ and sample the trapdoor key as*

$$tk = (r_1, \ldots, r_l) \leftarrow \mathbb{Z}_{|\mathbb{G}|}^l$$

and the injective key as

$$ik = \begin{pmatrix} \mathbf{g} \\ \mathbf{g}^{r_1} \\ \vdots \\ \mathbf{g}^{r_l} \end{pmatrix} \tag{3.7.1}$$

where $\mathbf{g} \leftarrow \mathbb{G}^l$.

- *F: on injective key*

$$ik = \begin{pmatrix} \mathbf{g}' \\ \mathbf{g}'_1 \\ \vdots \\ \mathbf{g}'_l \end{pmatrix} \tag{3.7.2}$$

and domain point $x \in \{0,1\}^l$ return

$$F_{ik}(x) = \begin{pmatrix} x \odot \mathbf{g}' \\ (x \odot \mathbf{g}'_1) \cdot g^{x_1} \\ \vdots \\ (x \odot \mathbf{g}'_1) \cdot g^{x_l} \end{pmatrix} \tag{3.7.3}$$

- *F^{-1}: on trapdoor key $(r_1, \ldots, r_l) \in \mathbb{Z}^l_{|G|}$ and image*

$$ik = \begin{pmatrix} g' \\ g'_1 \\ \vdots \\ g'_l \end{pmatrix} \tag{3.7.4}$$

return $x = x_1 x_2 \ldots x_l \in \{0,1\}^l$ where x_i is the bit such that

$$g'_i = (g')^{r_i} \cdot g^{x_i}$$

Next, we state a statement, implicit in [20], that shows a DDH-based encryption scheme that provides strong-(λ, l)-entropy circular security. See [57] on how this statement is derived from the result of [20].

Theorem 15. *(Implicit in [20]) Let $v = v(n)$ be an upper-bound on the size of any group output by $\mathcal{G}(1^n)$. Letting $\lambda = \log v + h$, where $h \in \omega(\log n)$ is an arbitrary function, and $l > \lambda$ be an arbitrary value, the scheme of Definition 14, when parameterized with l, is strongly-(λ, l)-entropy circular secure.*

3.7.2 From the Quadratic Residuosity (QR) and Related Assumptions

Brakerski and Goldwasser [23] construct a circular-secure encryption scheme (to which we refer as the BG scheme) from a general assumption that they call the *subgroup indistinguishability assumption*, which is in particular implied by the QR and Paillier's decisional composite residuosity (DCR) [79] assumptions. We show that the QR-based circular-secure bit-encryption scheme of Brakerski and Goldwasser satisfies the reproducibility property; the analyses for the other schemes follow similarly.

For an *RSA number* N (i.e., $N = pq$, where p and q are distinct odd primes) we use $\mathcal{QR}_N$ to denote the subset of $\mathbb{Z}_N^*$ consisting of quadratic residues modulo N, and let $\mathcal{J}_N$ denote the set of elements in $\mathbb{Z}_N^*$ with Jacobi symbol one. Finally, we define $\mathcal{QNR}_N = \mathcal{J}_N \setminus \mathcal{QR}_N$.

Assume that $RSAGen(1^n)$ is a PPT algorithm that on input 1^n generates a *Blum integer* N, i.e., $N = pq$ with p and q being distinct primes satisfying the condition $p, q \equiv 3 \pmod 4$. We say that the *quadratic residuosity* (QR) problem is hard under RSAGen if $\{N, U(\mathcal{QR}_N)\}_{n\in\mathbb{N}}$ is computationally indistinguishable from $\{N, U(\mathcal{QNR}_N)\}_{n\in\mathbb{N}}$, where N is generated according to $RSAGen(1^n)$.

We now describe the BG scheme.

Definition 15. *(From [23])*

- $Param(1^n)$*: returns* $(N, \mathbf{g})$*, where* $N \leftarrow RSAGen(1^n)$ *and* $\mathbf{g} \leftarrow \mathcal{QR}_N^l$*;*
- $Gen(1^n)$*: samples the secret key as* $sk \leftarrow \{0,1\}^l$ *and sets the public key* $pk = (sk \odot \mathbf{g})^{-1}$*;*
- $E_{pk}(b))$*: samples* $r \in \mathbb{Z}_{N^2}$ *and returns* $(\mathbf{g}^r, pk^r \cdot (-1)^b)$*; and*
- $D_{sk}((\mathbf{g}^r, pk^r \cdot (-1)^b)$*: clear.*

The proof of reproducibility of the scheme above follows exactly as in the proof of the BHHO scheme.

3.8 Conclusions and Open Problems

We gave generic constructions of several cryptographic primitives based on a general technique for de-randomizing reproducible bit-encryption schemes. For all the primitives we built it is already known that a black-box construction from CPA-secure

encryption alone is either impossible, or very difficult to find. We mention a few open problems that arise from our work. First, it would be interesting to see if the black-box result of [48] already separates TDFs from circular-secure encryption; showing this would imply that our reliance on an additional property, i.e., reproducibility, is unavoidable. Second, we would like to see whether the LWE-based circular-secure scheme of Applebaum *et al.* [4] can be used to instantiate our base assumptions. Finally, as mentioned earlier, our techniques allow us to understand better the relations between certain circular-secure schemes and DE-secure schemes. It would be interesting to see if similar connections could be proved in other settings. For example, DDH-based constructions of DE schemes satisfying *auxiliary-input security* [26] share certain design principles with those of randomized schemes satisfying *auxiliary-input leakage resilience* [35]; however, a generic connection is still not known.

Chapter 4

Black-Box Separation of (Seed-)Circular and Semantic Security

In this chapter we focus on the problem of understanding the black-box complexity of assumptions behind the notion of circular security. We begin with an introduction, stating informally the results proved in this chapter and also comparing the results with previous work.

4.1 Introduction

In this chapter we address the question of whether circular-secure bit encryption can be based on the minimal primitive of semantically-secure encryption. As mentioned earlier all constructions of circular-secure encryption in the literature are based on assumptions with certain algebraic properties [20, 4, 23, 94]. Moreover, the circular-secure construction that we presented in Chapter 2 is based on a public-key encryption scheme that meets two strong structural properties. This naturally raises the question that whether circular-secure encryption can be based on standard encryption schemes with no structure.

We first prove that there exists no fully black-box construction of *seed-circular-secure* bit encryption schemes from semantically-secure encryption schemes. A bit encryption scheme is seed-circular secure if it is semantically secure in the presence of an encrypted version of the seed (of the key-generation algorithm) under its corre-

sponding public key. Our proof indeed shows that CCA2-secure encryption does not imply *weak*-seed-circular-secure encryption: The latter notion weakens that of seed-circular-security by only guaranteeing that no adversary can compute the seed in full from an encrypted seed with non-negligible probability. The above result already rules out certain types of constructions for circular-secure encryption (e.g., those in which seeds and secret keys are the same). We will show how to adapt our results to the setting of circular security, to rule out a large and non-trivial class of constructions of circular-secure encryption that we call *key-isolating constructions*.

Discussion of notions. We mention a few points about the notion of seed-circular security. First, one may argue that this notion is in fact a stronger version of that of standard circular security, since seeds "contain" more information than secret keys, making it less secure for seeds to be encrypted under their corresponding public keys. This intuition is indeed formalizable for certain schemes. For example, if $\mathcal{E}$ is fully homomorphic (or homomorphic enough to evaluate G), then if $\mathcal{E}$ is seed-circular secure it is also circular secure, since one can use the homomorphic properties of $\mathcal{E}$ to evaluate G homomorphically, thereby producing an encrypted secret key from an encrypted seed. (This simple proof is, however, non-black-box.) From a practical point of view, the notion of seed circular security for specific schemes is not very natural since such schemes typically come with *public parameters* (e.g., a group), and it is not very meaningful to talk about, say, encrypting the bits used to generate those parameters. Nevertheless, if public-parameter generation is thought of as a separate process, many specific schemes have the property that their secret keys are just the same as their seeds (e.g., ElGamal). For example, both circular-secure schemes of [20, 23] have the property that with respect to fixed public parameters (which are a group plus l group elements), their secret keys are just random l-bit-strings, being the same as their seeds. Thus, as a step toward proving full black-box impossibility for circular-secure encryption, it may be worthwhile to formulate a notion of encryption with public parameters, and investigate whether our results extend to this case. We have not, however, carried this out at this moment.

Comparison with [55]. Haitner and Holesntein [55] rule out fully-black-box constructions of fully-KDM-secure schemes from trapdoor permutations, by giving a family F of $poly(n)$-wise independent hash functions for any proposed construction (G, E, D) and showing that breaking the F-KDM security of (G, E, D) cannot be reduced to inverting trapdoor permutations. One may wonder whether this combined

with Applebaum's result [3], which shows how to amplify projection security into F-KDM security (for any F), already separates projection security from trapdoor permutations. This does not seem to be the case. Roughly speaking, under [55] the $poly(n)$ parameter of F depends on the complexity of E, while under Applebaum's amplification the function family F' is first fixed, and based on F' a scheme is obtained whose, in particular, encryption function's complexity grow by that of F'. This subtle gap does not seem to be filled.

Comparison with [89]. The results of [89] show that no fully-black-box reduction can prove that any semantically-secure bit-encryption scheme is circular secure. Note that this result considers reductions between security properties of the *same* scheme; our results on the other hand focus on constructions.

4.1.1 Our Black-Box Separation Model

Our separation model follows that of Gertner *et al.* [48]. In particular, for any candidate seed-circular-secure construction $\mathcal{E} = (G, E, D)$ we show the existence of two oracles $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ and $\mathbf{T}$ such that (a) there exists a PPT oracle adversary $\mathcal{A}^{\mathbf{O},\mathbf{T}}$ that breaks the (supposed) seed-circular security of $\mathcal{E}^{\mathbf{O}}$ and (b) no PPT oracle adversary $\mathcal{B}^{\mathbf{O},\mathbf{T}}$ can break the semantic security of $\mathbf{O}$. This immediately implies that there exists no fully-black-box reduction. As common in separation models we show the existence of $\mathbf{O}$ and $\mathbf{T}$ non-constructively by proving results with respect to randomly chosen $\mathbf{O}$ and $\mathbf{T}$. We give an overview of our techniques and separation model in Section 4.5.

Most separation results in the literature indeed rule out the existence of *relativizing reductions*, e.g., [67, 91, 46, 21, 92], which constitute a broader class of constructions than fully-black-box ones. We stress that our results do not rule out relativizing reductions. Nonetheless, we are not aware of any "natural" cryptographic construction that is relativizing but not fully-black-box. Finally, we mention that there exists separation results in the literature that also only rule out fully-black-box reductions, e.g., [66, 55, 69].

4.1.2 Black-Box versus Non-Black-Box Techniques

In Section 1.1 we cited some non-black-box techniques in cryptography. Here we focus our discussion on non-black-box reductions that were later replaced with black-box

ones, or for which a black-box counterpart is still eluding or is known to be impossible. Here by non-black-box we mean a reduction with a non-black-box construction.

We mention [31, 68] as two examples of black-box constructions that replaced their earlier non-black-box counterparts [82, 51]. Classical examples of non-black-box constructions with no known black-box counterparts are [77, 38], giving non-black-box constructions of CCA1- and CCA2-secure encryption from *enhanced trapdoor permutations*. The state of our knowledge regarding the black-box status of CCA-secure encryption versus other "classical" public-key primitives is arguably limited, and the only known works are the work of Gertner *et al.* [47], ruling out *shielding* black-box constructions of CCA1-secure encryption from CPA-secure encryption, and that of Myers and Shelat [73] proving an equivalence between one-bit and many-bit CCA2 secure encryption. Finally, we mention that the work of Mahmoody and Pass [69] shows the existence of a non-black-box construction (that of non-interactive commitment schemes from so called *hitting one-way functions*) for which provably no black-box-counterpart exists.

4.2 Chapter Organization

The rest of the chapter is organized as follows. We first prove our separation result (based on semantic security) for seed circular security (Sections 4.3 to 4.9) and then in Section 4.10 we show how to adapt our results to obtain a partial separation for circular security. In Section 4.11 we highlight the main bottlenecks that prevented us from obtaining a full separation result for circular security. Finally, we conclude the discussion in Section 4.12 by mentioning a few open problems.

4.3 Preliminaries

The notion of a public-key encryption scheme (PKE) (G, E, D) was defined earlier. We refer to the randomness space of G as the *seed* space of the scheme. We assume the decryption algorithm is deterministic, and always decrypts correctly, and refer to this as the *correctness* condition. All schemes in this chapter are bit encryption schemes. Whenever we write $E(PK, S)$ for $S \in \{0,1\}^*$ it should be interpreted as the bit-by-bit encryption of S.

We shall use lowercase letters $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ to denote base (i.e., black-box) schemes and uppercase letters (G, E, D) to denote constructions.

Oracle convention. Whenever we talk about an oracle adversary/algorithm $\mathcal{A}$ we adopt the following conventions: we say $\mathcal{A}$ is *efficient* (or PPT) if $\mathcal{A}$ can be implemented as a PPT oracle algorithm; we say $\mathcal{A}$ is *query-efficient* if $\mathcal{A}$ always makes at most a poly-number of oracle queries (but unlimited otherwise, and may run exponential local computations). Whenever we put no restriction on an adversary it means that it is not restricted in any way.

Inequality convention. For two functions P_1 and P_2, whenever we write $P_1(n) \leq P_2(n)$ we mean that the inequality holds asymptotically, i.e., for all sufficiently large n.

We define when an adversary breaks the (seed-)circular security of a scheme.

Definition 16. *Let $\mathcal{E} = (G, E, D)$ be a PKE with seed space $\{0,1\}^n$. Let $S \leftarrow \{0,1\}^n$, $(PK, SK) = G(S)$, $b \leftarrow \{0,1\}$ and $C \leftarrow E(PK, b)$. We say that $\mathcal{A}$* breaks the seed-circular security of $\mathcal{E}$ *if $\Pr[\mathcal{A}(PK, E(PK, S), C) = b]$ is non-negligibly better than $1/2$. We say $\mathcal{A}$* breaks the weak-seed-circular security of $\mathcal{E}$ *if $\Pr[\mathcal{A}(PK, E(PK, S)) = S]$ is non-negligible. We say $\mathcal{A}$* breaks the circular security of $\mathcal{E}$ *if $\Pr[\mathcal{A}(PK, E(PK, SK), C) = b]$ is non-negligibly better than $1/2$.*

We define a notion of fully-black-box reductions between encryption primitives. See [86, 7] for more general notions of black-box reductions.

Definition 17. *A* fully-black-box reduction of *P-secure (e.g., circular-secure) encryption to Q-secure (e.g., CPA-secure) encryption consists of two PPT oracle algorithms $(\mathcal{E}, Red)$, satisfying the following: for any PKE $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$,*

1. *$\mathcal{E}^{\mathbf{O}} = (G^{\mathbf{O}}, E^{\mathbf{O}}, D^{\mathbf{O}})$ forms a PKE, and*
2. *for any adversary $\mathcal{A}$ breaking the P-security of $(G^{\mathbf{O}}, E^{\mathbf{O}}, D^{\mathbf{O}})$, the oracle algorithm $Red^{\mathcal{A},\mathbf{O}}$ breaks the Q-security of $\mathbf{O}$.*

4.4 PKE Oracle Distribution

Convention. Whenever we say a function $f\colon D \to R$ with property P (e.g., injectivity) is a randomly chosen function we mean f is chosen uniformly at random from the space of all functions from D to R having property P.

We describe a distribution under which a PKE oracle (with some auxiliary oracles) is sampled. We largely follow the notational style of [47]. As notation, if f is a

function whose output is a tuple, say a pair, we write $f(x) = (*, y)$ to indicate that $f(x) = (y', y)$, for some y'.

Definition 18. *We define an oracle distribution Ψ which produces a PKE oracle with certain length parameters, plus two auxiliary oracles. Formally, Ψ produces an ensemble of oracles $\mathcal{O}_n = (\mathbf{O}_n, \mathbf{u}_n, \mathbf{w}_n)_{n\in\mathbb{N}}$, where for every $n \in \mathbb{N}$, $\mathbf{O}_n = (\mathbf{g}_n, \mathbf{e}_n, \mathbf{d}_n)$ and $(\mathbf{u}_n, \mathbf{w}_n)$ are chosen as follows.*

- $\mathbf{g}_n \colon \{0,1\}^n \to \{0,1\}^{5n}$ *is a random one-to-one function, mapping a secret key to a public key.*
- $\mathbf{e}_n \colon \{0,1\}^{5n} \times \{0,1\} \times \{0,1\}^n \to \{0,1\}^{7n}$ *is a function, where for every $pk \in \{0,1\}^{5n}$, $\mathbf{e}_n(pk, \cdot, \cdot)$ is a random one-to-one function.*
- $\mathbf{d}_n \colon \{0,1\}^n \times \{0,1\}^{7n} \to \{0,1\} \cup \{\perp\}$ *is defined by letting $\mathbf{d}_n(sk, c) = b$ if and only if $\mathbf{e}_n(\mathbf{g}_n(sk), b, r) = c$, for some $r \in \{0,1\}^n$; otherwise, $\mathbf{d}_n(sk, c) = \perp$.*
- $\mathbf{u}_n \colon \{0,1\}^{5n} \times \{0,1\}^{7n} \to (\{0,1\} \times \{0,1\}^n) \cup \{\perp\}$ *is defined as $\mathbf{u}_n(pk, c) = (b, r)$ if $\mathbf{e}_n(pk, b, r) = c$, and $\mathbf{u}_n(pk, c) = \perp$ if for no (b, r) does it hold that $\mathbf{e}_n(pk, b, r) = c$. That is, $\mathbf{u}_n(pk, c)$ decrypts c relative to pk, and if successful, also returns the unique randomness used to produce c.*
- $\mathbf{w}_n \colon \{0,1\}^{5n} \to \{\perp, \top\}$ *is defined as $\mathbf{w}_n(pk) = \top$ if for some sk $\mathbf{g}_n(sk) = pk$, and $\mathbf{w}_n(pk) = \perp$, otherwise. That is, $\mathbf{w}_n(pk)$ checks whether pk is a valid public key.*

Query size. We define the *size* of a query q to $\mathbf{h}_{n'}$, where $\mathbf{h}$ is a function parameterized by a security parameter, to be n'. That is, the size of a query is not necessarily the same but poly-related to the bit-length of the query.

Omitting the security parameter. We define $\mathbf{g}(sk) = \mathbf{g}_n(sk)$, for every n and $sk \in \{0,1\}^n$, and use a similar convention for other functions in Definition 18. Sometimes when we need to emphasize under what security parameter a query is made, we put in the sub-index n; in other places we typically omit the sub-index.

Ψ-valid oracles. We call a triple of functions $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ *Ψ-valid* if $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ is part of a possible output of Ψ, i.e., the domains and ranges of $\mathbf{g}$, $\mathbf{e}$ and $\mathbf{d}$ are as specified in Definition 18, and also all the corresponding injectivity conditions hold. Similarly, we may use the same convention to call, say, $\mathbf{g}$, Ψ-valid.

Notation. For oracles $O = (O_1, \ldots, O_m)$ and an oracle algorithm $\mathcal{A}^O$, we let $qry = \langle O_i, q\rangle$ denote an $\mathcal{A}$'s query q to oracle O_i; if $u = O_i(q)$ we use $(\langle O_i, q\rangle, u)$ to indicate that $\mathcal{A}$ calls O_i on q and receives u; we also define $O(qry) = u$. If Que is a set of such *query/response pairs* we use shorthands like $(\langle O_j, *\rangle, u) \in \mathsf{Que}$ to mean that for some q, $(\langle O_j, q\rangle, u) \in \mathsf{Que}$. Thus, $(\langle O_j, *\rangle, u) \notin \mathsf{Que}$ indicates that for no q, we have $(\langle O_j, q\rangle, u) \in \mathsf{Que}$.

Symbolic representation of oracle queries. Sometimes we need to talk about sets containing query/response pairs generated under some oracle, and later on check them against another oracle. For this reason, we may sometimes talk about *symbolic* query/response pairs. For example, the symbolic form of a concrete query/response pair $(\langle \mathbf{g}, sk\rangle, pk)$ is denoted $(\langle \mathfrak{g}, sk\rangle, pk)$.

4.5 General Overview of the Separation Model and Techniques

Summary of approach. We start by giving a high-level overview of our separation approach. Let $\mathcal{E} = (G, E, D)$ be a candidate construction, $(\mathbf{O}, \mathbf{u}, \mathbf{w}) \leftarrow \Psi$ and $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$. Our goal is to give an oracle $\mathbf{T}$ s.t. (I) $\mathbf{T}$ is helpful in breaking the (alleged) seed-circular-security of $G^{\mathbf{O}}$ and (II) $\mathbf{T}$ not helpful in breaking the CPA security of $\mathbf{O}$. The most obvious idea is that on inputs of the form $(PK, C_1, \ldots, C_n)$, an alleged public key PK and an encryption of PK's seed (under $\mathcal{E}^{\mathbf{O}}(PK, \cdot)$), $\mathbf{T}$ will check whether PK is a valid public key under $G^{\mathbf{O}}$ and if so decrypt $C_1, \ldots, C_n$ under a secret key corresponding to PK to get S and return S if $G^{\mathbf{O}}(S)$ produces PK. There are two problems with this approach. First, even doing a simple check, namely whether PK is a valid public key, can potentially grant a CPA adversary against $\mathbf{O}$ much power. (It is not hard to think of contrived constructions $\mathcal{E}$ for which this is the case.) Second, even if we assume a CPA-adversary $\mathcal{A}^{\mathbf{O},\mathbf{T}}(pk, c)$ always calls $\mathbf{T}$ on valid PK's, we still have to make sure that $\mathcal{A}$ cannot come up with a clever query $\mathbf{T}(PK, C_1, \ldots, C_n)$ whose response leaks information about $\mathbf{g}^{-1}(pk)$ or about c's plaintext bit.

Our approach starts by first resolving Problem (1) using an idea from [48] (also used in some subsequent works [47, 92]): the oracle $\mathbf{T}$ performs the decryption of $(C_1, \ldots, C_n)$ not relative to $\mathbf{O}$, but relative to some $\widetilde{\mathbf{O}}$, under which PK is indeed a valid pair of public/secret keys (i.e., $\mathbf{T}$ decrypts using $D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n)$, where

$(PK, SK') \in G^{\widetilde{\mathbf{O}}}$). Without further restrictions on $\widetilde{\mathbf{O}}$ the result of decryption is most likely a random noise, as $\widetilde{\mathbf{e}}$ and $\widetilde{\mathbf{d}}$ can behave arbitrarily. Thus, we also need to ensure that with high probability $E^{\mathbf{O}}(PK, b; R) = E^{\widetilde{\mathbf{O}}}(PK, b; R)$, for any bit b. Specifically, we construct $\widetilde{\mathbf{O}}$ by *super-imposing* a polynomial number of query/response pairs, which serve as a *certificate* of PK's validity, on $\mathbf{O}$. To resolve Problem (2), the oracle $\mathbf{T}$ will refuse to decrypt queries which are deemed "dangerous": those that can be issued by a CPA adversary, and whose responses could potentially leak information about $\mathcal{A}$'s challenge secrets. The main challenge is to formulate these dangerous queries in such a way that $\mathbf{T}$ is provably of no use to any CPA adversary against $\mathbf{O}$, while guaranteeing that $\mathbf{T}$ still decrypts, with high probability, a randomly encrypted random seed.

Concrete overview. We now give a concrete overview of the above approach for a simplified construction. We will need the following definition and assumption.

Definition 19. *We define the following procedure we call KeyImpose.*

- ***Input****:* $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ *and a set* $\mathsf{Q_s} = \{(\langle \mathfrak{g}, sk_1 \rangle, pk_1), \ldots, (\langle \mathfrak{g}, sk_w \rangle, pk_w)\}$*, satisfying* $sk_i \neq sk_j$ *for any different* i *and* j*.*
- ***Output****:* $(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}})$*, where*

$$\widetilde{\mathbf{g}}(sk) = \begin{cases} \mathbf{g}(sk) & \text{if } sk \notin \{sk_1, \ldots, sk_w\} \\ pk_i & \text{if } sk = sk_i \text{ for some } 1 \leq i \leq w \end{cases} \tag{4.5.1}$$

  $\widetilde{\mathbf{d}}(sk, c)$ *is defined as follows: if there exist* b *and* r *such that* $\mathbf{e}(\widetilde{\mathbf{g}}(sk), b, r) = c$ *then* $\widetilde{\mathbf{d}}(sk, c) = b$*; otherwise,* $\widetilde{\mathbf{d}}(sk, c) = \perp$*.*

Note that in the above definition if $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ is a valid PKE scheme and $\mathsf{Q_s}$ satisfies the required condition then $(\widetilde{\mathbf{g}}, \mathbf{e}, \widetilde{\mathbf{d}})$ is also a valid PKE scheme.
We assume the following for any construction (G, E, D) discussed throughout.

Assumption 2. *For any* Ψ*-valid* $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$*, we assume* $G^{\mathbf{O}}$*,* $E^{\mathbf{O}}$ *and* $D^{\mathbf{O}}$*, on inputs corresponding to security parameter* 1^n*, make exactly* n^s *oracle calls (for* $s \geq 1$*), each of size at most* n^s*, and that* $G^{\mathbf{O}}(1^n)$ *uses exactly* n *random bits.*

We now describe our techniques for a simple class of constructions, those with oracle access of the form $(G^{\mathbf{g}}, E^{\mathbf{e}}, D^{\mathbf{d}})$. We first give the oracle $\mathbf{T}$, defined with respect to a fixed $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ and a fixed construction (G, E, D), which helps us to break

the seed-circular security of $(G^{\mathbf{g}}, E^{\mathbf{e}}, D^{\mathbf{d}})$. Fix (G, E, D) throughout this section, so we make the dependence of $\mathbf{T}$ on (G, E, D) implicit below. The oracle $\mathbf{T}$ is selected from a class of oracles, but it is convenient to define the output distribution of a randomly chosen $\mathbf{T}$ on an arbitrary given input, as we do below.

Description of T:
Oracles: $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{w})$
Input: $(1^n, PK, C_1, \ldots, C_n)$
Operations:

1. Choose $(\mathbf{g}', S')$ uniformly at random from the set of all pairs satisfying (a) $\mathbf{g}'$ is Ψ-valid and (b) $G^{\mathbf{g}'}(1^n, S') = (PK, *)$. If no such a pair exists return $\perp$. Otherwise, let SK' be the secret key output by $G^{\mathbf{g}'}(1^n, S')$.

2. Let $\mathsf{Q_s}$ contain the symbolic versions of all query/response pairs made in the execution of $G^{\mathbf{g}'}(1^n, S')$. Define $(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_s})$. Let QPub have any pk s.t. (a) $\mathbf{w}(pk) = \top$, (b) $|pk| \geq 5 \log n^{2s}$ and (c) $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Q_s}$.

3. Compute $S_{out} = D^{\widetilde{\mathbf{d}}}(SK', C_1 \ldots C_n)$. Execute $G^{\mathbf{g}}(S_{out})$ and if for all $pk \in \mathsf{QPub}$ the query/response $(\langle \mathbf{g}, * \rangle, pk)$ is made during the execution, then return S_{out}; otherwise, return $\perp$.

For our discussion below we need the following two lemmas, also used in later sections. Lemma 9 is used to show that $\mathbf{T}$ helps in breaking the seed-circular security of the construction, and Lemma 10 is used to argue $\mathbf{T}$ is not helpful in breaking the CPA security of a random $(\mathbf{g}, \mathbf{e}, \mathbf{d})$. As a terminology, we call an adversary $\mathcal{A}^{(\mathbf{u}, \ldots)}$ *CCA2-valid* if $\mathcal{A}$ on input (pk, c) never calls $\langle \mathbf{u}, (pk, c) \rangle$.

Lemma 9. *Let P be a function satisfying $P(n) \geq n$. Let $\mathcal{B}$ be an oracle adversary, which has access to some Ψ-valid oracle $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w})$, and which on input 1^n makes a list Que of at most $P = P(n)$ queries and outputs a public key pk_{out}, such that $|pk_{out}| \geq 5 \log P$. It then holds that*

$$\Pr_{\mathcal{O} \leftarrow \Psi}[\mathbf{w}(pk_{out}) = \top \text{ and } (\langle \mathbf{g}, * \rangle, pk_{out}) \notin \mathsf{Que}] \leq \frac{1}{P}.$$

Proof. The proof of the lemma uses simple probabilistic arguments so we omit most of the details. Assume $\mathcal{B}$ at the end of its computation calls $\mathbf{w}$ on the public key it outputs. This only increases the number of queries by one. At any point during $\mathcal{B}$'s

computation with current query list Que we say $\mathcal{B}$'s next query produces a *hit* if the next query is of the form $\langle \mathbf{w}, pk\rangle$ and it holds that $|pk| \geq 5\log P$, $\mathbf{w}(pk) = \top$ and $(\langle \mathbf{g}, *\rangle, pk) \notin \mathsf{Que}$. The adversary $\mathcal{B}$ wins if during $\mathcal{B}$'s computation at least one hit occurs. Assume the ith query is $\langle \mathbf{w}, pk\rangle$, for $|pk| = 5\log P_1 \geq 5\log P$. The probability that the ith query produces a hit given there were no previous hits is at most

$$\frac{P_1}{P_1^5 - P} \leq \frac{1}{P_1^3} \leq \frac{1}{P^3}.$$

In above we used the fact that the number of valid public keys of length $5\log P_1$ is P_1 and the number of all public keys of length $5\log P_1$ is P_1^5. Using a union bound, we may bound the probability of the theorem as $(P+1)/(P^3) \leq 1/P$. □

Lemma 10. *Let $\mathcal{B}$ be a CCA2-valid oracle adversary, which has access to some Ψ-valid oracle $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w})$, and which on input $(1^n, pk, c)$ makes at most $2^{n/4}$ queries and outputs a bit. It then holds that*

$$\Pr\left[\mathcal{B}^{\mathcal{O}}(1^n, pk, c) = b\right] \leq \frac{1}{2} + \frac{1}{2^{n/4}},$$

where $\mathcal{O} \leftarrow \Psi$, $b \leftarrow \{0,1\}$, $sk \leftarrow \{0,1\}^n$, $pk = \mathbf{g}(sk)$ and $c \leftarrow \mathbf{e}(pk, b)$.

Proof. The proof of the lemma uses simple probabilistic arguments so we omit most of the details. Let b_{out} be $\mathcal{A}(1^n, pk, c)$'s output and let Que be the set of $\mathcal{A}$'s query/response pairs at the end of the computation. For bit b' let $\mathsf{Known}_{b'}$ contain any c' such that $(\langle \mathbf{e}, (pk, b', *)\rangle, c') \in \mathsf{Que}$ or $(\langle \mathbf{u}, (pk, c')\rangle, \top)$. The set $\mathsf{Known}_{b'}$ contains all ciphertexts c' that $\mathcal{A}$ knows are encryptions of b' under its challenge public key pk. Without loss of generality assume that $\mathcal{A}$ makes exactly $2 \times 2^{n/4}$ queries but at the end $|\mathsf{Known}_0| = |\mathsf{Known}_1|$. (If $\mathcal{A}$ needs to, say, increase the size of Known_0 it can make queries of the form $\langle \mathbf{e}, (pk, 0, *)\rangle$ as many as needed.) Also, recall by Assumption 3 that any $\langle \mathbf{d}, (sk', *)\rangle$ query of $\mathcal{A}$ is preceded by $\langle \mathbf{g}, sk'\rangle$.

Let *pubhit* be the event that $(\langle \mathbf{g}, *\rangle, pk) \in \mathsf{Que}$ and let *ciphhit* be the event that $c \in \mathsf{Known}_0 \cup \mathsf{Known}_1$. By the facts that $\mathbf{g}_n \colon \{0,1\}^n \to \{0,1\}^{5n}$ and $\mathbf{e}_n \colon \{0,1\}^{5n} \times \{0,1\} \times \{0,1\}^n \to \{0,1\}^{7n}$, we can upperbound the probability of *pubhit* $\vee$ *ciphhit* as

$$\frac{2^{n/4+1}}{2^{5n}} + \frac{2^{n/4+1}}{2^n} \leq \frac{1}{2^{n/4+1}} + \frac{1}{2^{n/4+1}} = \frac{1}{2^{n/4}}. \tag{4.5.2}$$

Moreover, recalling that for Que it holds that $|\mathsf{Known}_0| = |\mathsf{Known}_1|$, we have

$$\Pr[b_{out} = b \mid \overline{pubhit} \wedge \overline{ciphhit}] = 1/2. \tag{4.5.3}$$

By combining Equations 4.5.2 and 4.5.3 the proof follows. □

We are now ready to discuss why $\mathbf{T}$ provides the "desired" properties. We remind the reader that the presentation in the rest of this section is informal.

$\mathbf{T}$ does not break the CPA security of $(\mathbf{g}, \mathbf{e}, \mathbf{d})$. We show that any adversary $\mathcal{A}^{\mathbf{O},\mathbf{T}}(1^n, pk, c)$, against the CPA-security of $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ for any Ψ-valid $(\mathbf{O}, \mathbf{u}, \mathbf{w})$, can be simulated by a CCA2-valid adversary $\mathcal{B}^{\mathbf{O},\mathbf{u},\mathbf{w}}(1^n, pk, c)$ that makes a poly-related number of queries. By Lemma 10 we then obtain our desired result. The crux of our techniques lies in showing how $\mathcal{B}$, using $\mathbf{u}$ and $\mathbf{w}$, can simulate $\mathcal{A}$'s access to $\mathbf{T}$.

$\mathcal{B}^{\mathbf{O},\mathbf{u},\mathbf{w}}(1^n, pk, c)$ starts running $\mathcal{A}^{\mathbf{O},\mathbf{T}}(1^n, pk, c)$ and forwards all $\mathcal{A}$'s $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ queries to its own corresponding oracles.

To respond to a $\mathbf{T}$ query of the form $Tqu \stackrel{\text{def}}{=} \langle \mathbf{T}, (1^{n_1}, PK, C_1, \ldots, C_{n_1}) \rangle$ made by $\mathcal{A}$, $\mathcal{B}$ acts as follows (note it may be that $n_1 \neq n$, as $\mathcal{A}$ can make queries under different security parameters): $\mathcal{B}$ forms SK' and $\mathsf{Q_s}$ exactly as in Steps 1 and 2 of $\mathbf{T}$'s computation. It is able to do so since during these two computations no queries are made to the real oracles (though, a massive offline search is involved). Next, $\mathcal{B}$ starts simulating $D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C_{n_1})$, where $(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_s})$. Since it is not clear how $\mathcal{B}$ can perform this decryption by only making a polynomial number of queries and without ever calling $\langle \mathbf{u}, (pk, c) \rangle$, we consider two possible cases:

(A) $(\langle \mathfrak{g}, * \rangle, pk) \notin \mathsf{Q_s}$: In this case we claim that $\mathcal{B}$ can successfully execute the decryption $D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C_{n_1})$. The reason is that for any query $qu = \langle \widetilde{\mathbf{d}}, (sk', c') \rangle$ during the execution, either (i) $(\langle \mathfrak{g}, sk' \rangle, *) \notin \mathsf{Q_s}$ or (ii) for some $pk' \neq pk$, $(\langle \mathfrak{g}, sk' \rangle, pk') \in \mathsf{Q_s}$. If (i) holds then $\widetilde{\mathbf{d}}(sk', c') = \mathbf{d}(sk', c')$ and so $\mathcal{B}$ can reply to qu by calling $\langle \mathbf{d}, (sk', c') \rangle$. If (ii) holds, then the answer to qu can be determined by calling $\langle \mathbf{u}, (pk', c') \rangle$, which is a valid query for $\mathcal{B}$ as $pk' \neq pk$. Now if $S_{out} = D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C_{n_1})$, then $\mathcal{B}$ performs the rest of Step 3 of $\mathbf{T}$, which $\mathcal{B}$ can successfully do since the rest only involves making $\mathbf{g}$ and $\mathbf{w}$ queries. Thus, $\mathcal{B}$ can find the answer to Tqu.

(B) $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Q_s}$: In this case, recalling the definition of Qpub, we have $pk \in \mathsf{QPub}$, since $\mathbf{w}(pk) = \top$ and $|pk| \geq 5 \log n_1^{2s}$. (The former is because pk is $\mathcal{B}$'s challenge public key which by definition is valid, and the latter is because

$n_1 = poly(n)$.) Thus, by the condition given in Step 3 of $\mathbf{T}$'s description, if $S_{out} = D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C_{n_1})$ then at least one of the following holds:

(a) The answer to Tqu is $\perp$; or

(b) The query/response pair $(\langle \mathbf{g}, * \rangle, pk)$ will show up during $G^{\mathbf{g}}(S_{out})$, i.e., $\mathbf{g}^{-1}(pk)$, $\mathcal{B}$'s challenge secret key, is revealed during $G^{\mathbf{g}}(S_{out})$.

We now claim that $\mathcal{B}$ *can find two strings* S_0 *and* S_1 *such that* $S_{out} \in \{S_0, S_1\}$. If this is the case, $\mathcal{B}$ can execute both $G^{\mathbf{g}}(S_0)$ and $G^{\mathbf{g}}(S_1)$; if during either execution a query/response $(\langle \mathbf{g}, * \rangle, pk)$ is observed, $\mathcal{B}$ has learned $\mathbf{g}^{-1}(pk)$, winning the CCA2 game; otherwise, $\mathcal{B}$ in response to Tqu returns $\perp$, which is indeed the correct response.

It remains to demonstrate the claim. To find S_0 and S_1, $\mathcal{B}$ attempts to simulate $D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C_{n_1})$. For any query $qu = \langle \widetilde{\mathbf{d}}, (sk', c') \rangle$ encountered in the simulation, one of the following holds

(i) $(\langle \mathfrak{g}, sk' \rangle, *) \notin \mathsf{Q_s}$; or

(ii) $(\langle \mathfrak{g}, sk' \rangle, pk') \in \mathsf{Q_s}$ for $pk' \neq pk$; or

(iii) $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$ and $c' \neq c$; or

(iv) $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$ and $c' = c$.

For Cases (i), (ii) and (iii) $\mathcal{B}$ can find the answer to qu by querying $\langle \mathbf{d}, (sk', c') \rangle$ for Case (i), querying $\langle \mathbf{u}, (pk', c') \rangle$ for Case (ii), and querying $\langle \mathbf{u}, (pk, c') \rangle$ for Case (iii). The latter two are legitimate $\mathbf{u}$ queries.

For Case (iv) $\mathcal{B}$ continues the execution of $D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C_{n_1})$ in two parallel branches BR_0 and BR_1, where $\mathcal{B}$ replies to qu with b_1 on BR_{b_1}. On both branches $\mathcal{B}$ replies to queries for which Cases (i), (ii) and (i) hold exactly as above. If on some branch $BR_{b'}$, still during the execution of $D^{\widetilde{\mathbf{d}}}(SK', C_1, \ldots, C'_{n_1})$, for a query qu' Case (iv) holds again (i.e., $qu' = \langle \widetilde{\mathbf{d}}, (sk', c) \rangle$ and $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$) $\mathcal{B}$ replies to qu' with b', making it consistent with the previous reply. Thus, these two branches result in two strings satisfying the claim.

By invoking Lemma 10 we deduce that for random $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ and $\mathbf{T}$, any $\mathcal{A}^{\mathbf{O},\mathbf{T}}(pk, c)$ that makes at most, say, $2^{n/5}$ queries (basically any number m of queries where $2^{n/4}/m$ is super-polynomial) has advantage at most $\frac{1}{2} + \frac{1}{2^{n/4}}$ of computing b, where $sk \leftarrow \{0,1\}^n$, $pk = \mathbf{g}(sk)$ and $c \leftarrow \mathbf{e}(pk, b)$.

T breaks seed-circular security: We show that $\mathbf{T}$ is useful if used honestly: For $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w}) \leftarrow \Psi$, $S \leftarrow \{0,1\}^n$, $(PK, SK) = G^{\mathbf{g}}(S)$ and $(C_1, \ldots, C_n) \leftarrow E^{\mathbf{e}}(PK, S)$, the probability that $\mathbf{T}(1^n, PK, C_1, \ldots, C_n)$ does not return S is at most $\frac{1}{n^2}$.

First, we claim that $S_{out} = S$, where S_{out} is the string constructed in Step 3 of $\mathbf{T}(1^n, PK, C_1, \ldots, C_n)$. This follows since (a) $(\widetilde{\mathbf{g}}, \mathbf{e}, \widetilde{\mathbf{d}})$ is a correct scheme (which can easily be checked), and (b) $(PK, SK') \in G^{\widetilde{\mathbf{g}}}(1^n)$. (Recall that SK' is constructed in Step 1 of $\mathbf{T}$.) Thus, by the correctness of the construction (G, E, D), $S_{out} = S$. Thus, $\mathbf{T}(1^n, PK, C_1, \ldots, C_n)$ either returns S or $\perp$.

Let $Fail$ be the event $\mathbf{T}(PK, C_1, \ldots, C_n) = \perp$ and $\alpha = \Pr[Fail]$. By definition $Fail$ is the following event: there exists pk s.t. (a) $|pk| \geq 5 \log n^{2s}$, (b) $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Q_s}$, (c) $\mathbf{w}(pk) = \top$ and (d) the query/response $(\langle \mathbf{g}, * \rangle, pk)$ does not show up during $G^{\mathbf{g}}(1^n, S)$. We show how to build an adversary $\mathcal{A}^{\mathcal{O}}$ that wins with probability α in the sense of Lemma 9 for $P = n^{2s}$. This shows $\alpha \leq \frac{1}{n^{2s}} \leq \frac{1}{n^2}$.

$\mathcal{A}^{\mathcal{O}}(1^n)$ generates $S \leftarrow \{0,1\}^n$ and $(PK, SK) = G^{\mathbf{g}}(S)$; it then samples a random $(\mathbf{g}', S')$ s.t. $G^{\mathbf{g}'}(S') = PK$ and lets $\mathsf{Q_s}$ contain the symbolic versions of all query/response pairs made to $\mathbf{g}'$. Denoting by Que the set of all query/response pairs of $\mathcal{A}$ so far (which was populated only during $G^{\mathbf{g}}(S)$), for all $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Q_s}$ such that $|pk| \geq 5 \log n^{2s}$ and that $(\langle \mathbf{g}, * \rangle, pk) \notin \mathsf{Que}$, $\mathcal{A}$ calls $\langle \mathbf{w}, pk \rangle$: as soon as $\mathcal{A}$ receives $\top$ in response, it returns pk. $\mathcal{A}$ makes at most $n^s + n^s \leq n^{2s}$ queries (see Assumption 2). Also, it is easy to see that $\mathcal{A}$ wins with probability α.

We can now, using standard techniques, combine the two facts above about $\mathbf{T}$ to rule out fully-black-box reductions for the construction type considered.

4.6 The Oracle T for (Breaking) Seed-Circular Security

In this section we describe the oracle $\mathbf{T}$ for general constructions.

Intuition. As in the previous section the main idea is to have the oracle $\mathbf{T}$, on input $(PK, C_1, \ldots, C_n)$, decrypt $C_1, \ldots, C_n$ relative to some $\widetilde{\mathbf{O}} = (\widetilde{\mathbf{g}}, \widetilde{\mathbf{e}}, \widetilde{\mathbf{d}})$, satisfying (a) $G^{\widetilde{\mathbf{O}}}$ produces $(PK, *)$ and (b) for any b with high probability $E^{\mathbf{O}}(PK, b, R) = E^{\widetilde{\mathbf{O}}}(PK, b, R)$. To obtain $\widetilde{\mathbf{O}}$ we may be tempted to proceed exactly as before, by sampling a set of query/response pairs Q and a seed S' such that $G^{\mathsf{Q}}(S') = (PK, *)$ and then *superimposing* Q (which now has all types of queries) on $\mathbf{O}$. While the

resulting $\widetilde{\mathbf{O}}$ satisfies Condition (a) it is not clear if Condition (b) is satisfied: The problem is there may be queries q asked quite frequently during random executions of $E^{\mathbf{O}}(PK, b)$ (call them *heavy*), and which may also occur in Q and receive a different response there. To overcome this problem we first run $E^{\mathbf{O}}(PK, b)$ for $b = 0, 1$ many times and collect all observed query/response pairs in a set Freq. We then force the sampled set Q to be *consistent* with Freq. Finally, we show how to superimpose Q on $\mathbf{O}$ to obtain $\widetilde{\mathbf{O}}$.

Setting things up. Fix the proposed construction (G, E, D). We now give an assumption to make our analysis easier and then give definitions formalizing the steps sketched above. We then use these definitions to define the oracle $\mathbf{T}$.

4.6.1 Definitions Underlying the Description of T

Assumption 3. *We assume that every oracle algorithm that has access to both* $\mathbf{g}$ *and* $\mathbf{d}$ *always queries* $\langle \mathbf{g}, sk \rangle$ *before querying* $\langle \mathbf{d}, (sk, *) \rangle$*. Also, we assume w.l.o.g. that* G *never calls the decryption algorithm of the base scheme,* $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$*. For ease of notation we keep* $\mathbf{d}$ *as a superscript to* G *and write* $G^{\mathbf{O}}$*.*

Definition 20. *We define the following probabilistic procedure, FreqQue.*

- ***Oracles:*** $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u})$
- ***Input:*** *A security parameter* 1^n *(left implicit), public key* PK *and* $p \in \mathbb{N}$*.*
- ***Output:*** *A set* Freq *formed as follows. For both* $b = 0, 1$ *run* $E^{\mathbf{O}}(1^n, PK, b)$ *independently* p *times and add the symbolic versions of all query/response pairs to* Freq*. Moreover, for any* $(\langle \mathfrak{d}, (sk, c) \rangle, *) \in \mathsf{Freq}$ *if* $\mathbf{u}(\mathbf{g}(sk), c) = (b', r') \neq \perp$ *add* $(\langle \mathfrak{e}, (pk, b', r') \rangle, c)$ *to* Freq*.*

  Note that by Assumption 2 $|\mathsf{Freq}| \leq 2pn^s + 2pn^s = 4pn^s$*.*

In the above definition apart from the actual observed query/response pairs we also enhanced Freq with some pairs obtained based on $(\langle \mathfrak{d}, (sk, c) \rangle, *)$ query/response pairs. This enhancement is only made to make some of the proofs simpler.

We say that oracle $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ is *consistent* with (or *agrees with*) a symbolic query/response pair $(\langle \mathfrak{g}, sk \rangle, pk)$ if $\mathbf{g}(sk) = pk$. The same definition can be given for other types of query/response pairs. We say $\mathbf{O}$ is consistent with a set of query/response pairs if $\mathbf{O}$ agrees with each element in the set.

Definition 21. *(Sampling consistent oracles) We define the following procedure we call ConsOrc.*

- ***Input****: a public key PK and a set Freq of symbolic query/response pairs.*
- ***Output****: a secret key SK' and query/response sets $\mathsf{Q_s}$, $\mathsf{Q_c}$ sampled as follows.*
  - *Sample $(\mathbf{g}', \mathbf{e}', \mathbf{d}', S')$ uniformly at random under the constraints that $\mathbf{O}' = (\mathbf{g}', \mathbf{e}', \mathbf{d}')$ is Ψ-valid and is consistent with Freq and that $G^{\mathbf{O}'}(S') = (PK, *)$. If no such a tuple exists, return $\perp$.*
  - *Let SK' be the secret-key outputted by $G^{\mathbf{O}'}(S')$ and let the sets $\mathsf{Q_s}$ and $\mathsf{Q_c}$ contain, respectively, the symbolic versions of all query/response pairs made to $\mathbf{g}'$ and $\mathbf{e}'$. (Recall by Assumption 3 no $\mathbf{d}'$-query is made.)*

We define the task of superimposing a set $\mathsf{Q_c}$ of $\mathfrak{e}$ queries on $(\mathbf{g}, \mathbf{e}, \mathbf{d})$: the result will be $(\mathbf{e}_{imp}, \mathbf{d}_{imp})$, perturbed versions of $(\mathbf{e}, \mathbf{d})$. Intuitively, we want $(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp})$ to form a PKE, $\mathbf{e}_{imp}$ to agree with $\mathsf{Q_c}$ and $(\mathbf{e}_{imp}, \mathbf{d}_{imp})$ to agree as much as possible with $(\mathbf{e}, \mathbf{d})$.

Definition 22. *We define the following procedure we call EncImpose.*

- ***Input****: a Ψ-valid $(\mathbf{g}, \mathbf{e}, \mathbf{d})$ and a set*

$$\mathsf{Q_c} = \{(\langle \mathfrak{e}, (pk_1, b_1, r_1)\rangle, c_1), \ldots, (\langle \mathfrak{e}, (pk_p, b_p, r_p)\rangle, c_p)\},$$

  Note that pk_i's above are not necessarily distinct.

- ***Output****: $(\mathbf{e}_{imp}, \mathbf{d}_{imp})$, defined as follows. First, let $\mathsf{W} = \{(pk_1, c_1), \ldots, (pk_p, c_p)\}$ and $\mathsf{W}' = \{(pk_1, \mathbf{e}(pk_1, b_1, r_1)), \ldots, (pk_p, \mathbf{e}(pk_p, b_p, r_p))\}$. Define*

$$\mathbf{e}_{imp}(pk, b, r) = \begin{cases} c_i & \text{if } (pk, b, r) = (pk_i, b_i, r_i), \text{ for some } 1 \le i \le p \\ \hat{c} & \text{if } (pk, \mathbf{e}(pk, b, r)) \in \mathsf{W} \\ \mathbf{e}(pk, b, r) & \text{otherwise} \end{cases} \tag{4.6.1}$$

  where the ciphertext $\hat{c}$ is defined as follows: Letting x be the smallest integer such that $(pk, \mathbf{e}(pk, b, r + x)) \notin \mathsf{W} \cup \mathsf{W}'$ we set $\hat{c} = \mathbf{e}(pk, b, r + x)$. Here, $r + x$ is done using a standard method.

$$\mathbf{d}_{imp}(sk,c) = \begin{cases} b_i & \text{if } \mathbf{g}(sk) = pk_i \text{ and } c = c_i \text{ for some } 1 \le i \le p \\ \mathbf{d}(sk,c) & \text{otherwise} \end{cases} \tag{4.6.2}$$

We justify the second case of $\mathbf{e}_{imp}$'s definition: if $(pk, \mathbf{e}(pk,b,r)) \in \mathsf{W}$, say that $(pk, \mathbf{e}(pk,b,r)) = (pk_i, c_i)$, we cannot set $\mathbf{e}_{imp}(pk,b,r) = \mathbf{e}(pk,b,r)$ as we have already set $c_i = \mathbf{e}_{imp}(pk_i, b_i, r_i)$: in particular, $\mathbf{e}_{imp}$ will be rendered incorrect if $b_i \neq b$. Thus, we keep shifting $\mathbf{e}(pk,b,r)$ (by adding x to r) until we hit a ciphertext $\hat{c}$ s.t. $(pk,\hat{c}) \notin \mathsf{W} \cup \mathsf{W}'$. The requirement $(pk,\hat{c}) \notin \mathsf{W}'$ is stronger than necessary, but will simplify some proofs. Note $\mathbf{e}_{imp}$ is not necessarily injective.

4.6.2 Description of the Weakening Oracle T

Description of T. We define the oracle $\mathbf{T}$. We first describe the output distribution of a random $\mathbf{T}$ on a single input-call, $(1^n, PK, C_1, \ldots, C_n)$, and then describe the underlying distribution from which $\mathbf{T}$ is chosen.
Oracles: $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w})$. Denote $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$.
Input: $(1^n, PK, C_1, \ldots, C_n)$

Operations:

1. **Learning frequent queries:** Let $\mathsf{Freq} \leftarrow FreqQue^{\mathbf{O},\mathbf{u}}(PK, n^{23s})$. Define $\mathsf{FreqPub}$ to be the set of public keys pk such that $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Freq}$.

2. **Sampling oracle/secret-key consistent with PK and Freq:** Sample

$$(SK', \mathsf{Q_s}, \mathsf{Q_c}) \leftarrow ConsOrc(PK, \mathsf{Freq}). \tag{4.6.3}$$

3. **Defining intermediate oracles:** Define

$$(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_c})$$
$$(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s}).$$

   Let $\widetilde{\mathbf{e}} = \mathbf{e}_{imp}$, and $\widetilde{\mathbf{O}} = (\widetilde{\mathbf{g}}, \widetilde{\mathbf{e}}, \widetilde{\mathbf{d}})$. Let QPub contain any pk such that (a) $\mathbf{w}(pk) = \top$, (b) $|pk| \ge 5 \log n^{25s}$ and (c) $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Q_s}$.

4. **Decrypting the encrypted input:** Compute $S_{out} = D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n)$.

5. **Returning S_{out} subject to a check:** Run $G^{\mathbf{O}}(S_{out})$ and let $\mathsf{EmbedPub}$ contain any pk such that the query/response $(\langle \mathbf{g}, * \rangle, pk)$ is made during $G^{\mathbf{O}}(S_{out})$. If $\mathsf{QPub} \subseteq \mathsf{EmbedPub} \cup \mathsf{FreqPub}$ return S_{out}; else, return $\perp$.

Notation. $Tvars^{\mathcal{O}}(PK)$ denotes the random variable $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}})$ obtained in the execution of $\mathbf{T}$ above with respect to $\mathcal{O}$ and PK. Note none of these random variables depend on $(C_1, \ldots, C_n)$. For the reader's convenience, we provide a table summary of how all these variables sampled on Page 105.

Remark about T. Note that the only part of the oracle $\mathbf{T}$ that involves making random choices are Step 1 (sampling from $FreqQue^{\mathbf{O},\mathbf{u}}(PK, n^{23s})$) and Step 2 (sampling from $ConsOrc(PK, \mathsf{Freq})$). The number of random coins required to do the sampling in Step 1 is obviously finite. For Step 2 recall that the output of $ConsOrc(PK, \mathsf{Freq})$ is formed based on sampling a Ψ-valid random oracle $\mathbf{O}'$ that is consistent with Freq and also that $G^{\mathbf{O}'}(1^n)$ generates PK (based on some seed). By default, $\mathbf{O}'$ should be defined for all security parameters. However, by Assumption 2 it suffices to sample $\mathbf{O}'$ only for security parameters up to n^s. Thus, for any fixed input $(1^n, PK, C_1, \ldots, C_n)$, the amount of randomness used by a random $\mathbf{T}$ to compute $\mathbf{T}(1^n, PK, C_1, \ldots, C_n)$ is finite.

Sampling space of T. We now explain how to choose a random $\mathbf{T}$. In particular, we would like a randomly chosen $\mathbf{T}$, if queried under a single input many times, to always return the same output. To this end, every possible $\mathbf{T}$ comes with a collection of random-coin strings, where for every possible query $qu = (1^n, PK, C_1, \ldots, C_n)$ to $\mathbf{T}$, the collection has a corresponding random-coin string $Coin_{qu}$, used by $\mathbf{T}$ to make the random choices that appear during the computation of $\mathbf{T}(qu)$. When we write $\Pr_{\mathbf{T}}[]$ we mean the probability is computed over a $\mathbf{T}$ chosen uniformly at random from the above-mentioned space.

$\mathsf{Freq} \leftarrow FreqQue^{\mathbf{O},\mathbf{u}}(PK, n^{23s})$ is sampled as follows.

1. For $b = 0, 1$: run $E^{\mathbf{O}}(PK, b)$ n^{23s} times and add the symbolic versions of all query/response pairs to Freq.
2. For any $(\langle \mathfrak{d}, (sk, c)\rangle, *) \in \mathsf{Freq}$ if $\mathbf{u}(\mathbf{g}(sk), c) = (b', r') \neq \perp$ add $(\langle \mathfrak{e}, (pk, b', r')\rangle, c)$ to Freq.

$(SK', \mathsf{Q_s}, \mathsf{Q_c}) \leftarrow ConsOrc(PK, \mathsf{Freq})$ is sampled as follows.

1. Sample $(\mathbf{g}', \mathbf{e}', \mathbf{d}', S')$ under the constraints that $\mathbf{O}' = (\mathbf{g}', \mathbf{e}', \mathbf{d}')$ is Ψ-valid and is consistent with Freq and that $G^{\mathbf{O}'}(S') = (PK, *)$.
2. Let SK' be the secret-key output of $G^{\mathbf{O}'}(S')$ and let $\mathsf{Q_s}$ and $\mathsf{Q_c}$ contain, respectively, the symbolic versions of all the query/response pairs made to $\mathbf{g}'$ and $\mathbf{e}'$.

$(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_c})$ is constructed as follows.

1. Assuming $\mathsf{Q_c} = \{(\langle \mathfrak{e}, (pk_1, b_1, r_1)\rangle, c_1), \ldots, (\langle \mathfrak{e}, (pk_p, b_p, r_p)\rangle, c_p)\}$ let $\mathsf{W} = \{(pk_1, c_1), \ldots, (pk_p, c_p)\}$.
2. Let $\mathsf{W}' = \{(pk_1, \mathbf{e}(pk_1, b_1, r_1)), \ldots, (pk_p, \mathbf{e}(pk_p, b_p, r_p))\}$
3. $$\mathbf{e}_{imp}(pk, b, r) = \begin{cases} c_i & \text{if } (pk, b, r) = (pk_i, b_i, r_i) \\ \hat{c} & \text{if } (pk, \mathbf{e}(pk, b, r)) \in \mathsf{W} \\ \mathbf{e}(pk, b, r) & \text{otherwise} \end{cases}$$

   where $\hat{c}$ is formed as follows: If x is the smallest integer s.t. $(pk, \mathbf{e}(pk, b, r + x)) \notin \mathsf{W} \cup \mathsf{W}'$, form $\hat{c} = \mathbf{e}(pk, b, r + x)$.

   $$\mathbf{d}_{imp}(sk, c) = \begin{cases} b_i & \text{if } \mathbf{g}(sk) = pk_i \text{ and } c = c_i \text{ for } i \leq p \\ \mathbf{d}(sk, c) & \text{otherwise} \end{cases}$$

Let $\widetilde{\mathbf{e}} = \mathbf{e}_{imp}$.The oracles $(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s})$ are built as follows.

1. $$\widetilde{\mathbf{g}}(sk) = \begin{cases} \mathbf{g}(sk) & \text{if } sk \notin \{sk_1, \ldots, sk_w\} \\ pk_i & \text{if } sk = sk_i \text{ for some } 1 \leq i \leq w \end{cases} \quad (4.6.4)$$
2. $\widetilde{\mathbf{d}}(sk, c)$ is defined as follows: if there exist b and r such that $\mathbf{e}(\widetilde{\mathbf{g}}(sk), b, r) = c$ then $\widetilde{\mathbf{d}}(sk, c) = b$; otherwise, $\widetilde{\mathbf{d}}(sk, c) = \perp$.

Table 4.1: Summary of $Tvars$ variables w.r.t. PK and $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w})$

4.7 T Breaks the Seed-Circular Security of (G, E, D)

We show if $\mathbf{T}$ is called honestly (i.e., on a random public key and a random encryption of the underlying seed) it will return the seed with high probability. To formalize the statement we define the following *environment* that specifies a random choice of $(\mathbf{O}, \mathbf{u}, \mathbf{w})$ plus those underlying an honest random input to $\mathbf{T}$.

Environment: $Env(n)$: Output $(\mathbf{O}, \mathbf{u}, \mathbf{w}, S, PK, C_1, \ldots, C_n)$, where:

1. $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w}) \leftarrow \Psi$ and $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$;
2. $S \leftarrow \{0,1\}^n$, $(PK, SK) \leftarrow G^{\mathbf{O}}(S)$ and $(C_1, \ldots, C_n) \leftarrow E^{\mathbf{O}}(PK, S)$.

Convention. Sometimes that we are interested only in a specific part of the output of $Env(n)$ we may use notation such as $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK) \leftarrow Env(n)$.

The following theorem shows $\mathbf{T}$'s usefulness in breaking seed-circular security.

Theorem 16. *It holds that*

$$\Pr_{\mathbf{Env},\mathbf{T}}[\mathbf{T}(PK, C_1, \ldots, C_n) = S] \geq 1 - \frac{1}{n^5}, \tag{4.7.1}$$

where $\mathbf{Env} = (\mathbf{O}, \mathbf{u}, \mathbf{w}, S, PK, C_1, \ldots, C_n) \leftarrow Env(n)$.

Proof layout. The proof consists of two parts. First, we show (Lemma 11) that with high probability $S_{out} = S$, where S_{out} is the string decoded in Step 4 of the execution of $\mathbf{T}(PK, C_1, \ldots, C_n)$. Next we show, conditioned on $S_{out} = S$, the probability that $\mathbf{T}(PK, C_1, \ldots, C_n)$ outputs $\perp$ is small (Lemma 12).

Lemma 11. *It holds*

$$\alpha(n) = \Pr[D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n) \neq S] \leq \frac{1}{2n^5},$$

where the probability is taken over the choices of $(\mathbf{O}, \mathbf{u}, \mathbf{w}, S, PK, C_1, \ldots, C_n) \leftarrow Env(n)$ *and* $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \leftarrow Tvars^{\mathbf{O},\mathbf{u},\mathbf{w}}(PK)$.

Lemma 12. *It holds*

$$\alpha(n) \stackrel{def}{=} \Pr_{\mathbf{Env},\mathbf{T}}\left[\left(D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n) = S\right) \wedge (\mathbf{T}(PK, C_1, \ldots, C_n) = \perp)\right] \leq \frac{1}{n^{25}},$$

where $\mathbf{Env} = (\mathbf{O}, \mathbf{u}, \mathbf{w}, S, PK, C_1, \ldots, C_n) \leftarrow Env(n)$, *and* SK' *and* $\widetilde{\mathbf{O}}$ *are the random variables sampled inside* $\mathbf{T}(PK, C_1, \ldots, C_n)$.

In what follows we show how to derive Theorem 16 from Lemmas 11 and 12. We prove Lemma 11 in Subsection 4.7.1 and Lemma 12 in Subsection 4.7.2.

Proof of Theorem 16. Let $Evnt$ be the event that $\mathbf{T}(PK, C_1, \ldots, C_n) \neq S$. We show that

$$\Pr_{\mathbf{Env},\mathbf{T}}[Evnt] \leq \frac{1}{n^5}. \tag{4.7.2}$$

We define the following two events:

- $Evnt1$: the event that $D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n) \neq S$.
- $Evnt2$: the event that

$$(D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n) = S) \wedge (\mathbf{T}(PK, C_1, \ldots, C_n) = \perp)$$

We claim

$$\Pr_{\mathbf{Env},\mathbf{T}}[Evnt] \leq \Pr_{\mathbf{Env},\mathbf{T}}[Evnt1 \vee Evnt2].$$

Note that by Lemmas 11 and 12 the above claim immediately implies

$$\Pr_{\mathbf{Env},\mathbf{T}}[Evnt] \leq \frac{1}{2n^5} + \frac{1}{n^{25}} \leq \frac{1}{n^5},$$

as desired. Thus, we show why our claim holds. To this end, we show whenever $Evnt$ holds, also $Evnt1 \vee Evnt2$ holds; or equivalently, if $\overline{Evnt1} \wedge \overline{Evnt2}$ holds, then $\overline{Evnt}$ holds. If $\overline{Evnt1} \wedge \overline{Evnt2}$ holds then

(a) $D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n) = S$; and

(b) $\mathbf{T}(PK, C_1, \ldots, C_n) \neq \perp$.

By the way $\mathbf{T}$ is designed if (a) and (b) hold then we definitely have $\mathbf{T}(PK, C_1, \ldots, C_n) = S$, namely $\overline{Evnt}$ holds. The proof is now complete. □

4.7.1 Proof of Lemma 11

We start with a simple fact: Informally, it states, in particular, that the string SK' built during an execution of $\mathbf{T}(PK, C_1, \ldots, C_n)$ is a matching secret key of PK relative to $G^{\widetilde{\mathbf{O}}}$.

Fact 1. *For any* $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ *and any* $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \in Tvars^{\mathcal{O}}(PK)$, *(a)* $\widetilde{\mathbf{O}}$ *is a correct PKE, and (b)* $(PK, SK') \in G^{\widetilde{\mathbf{O}}}(1^n)$.

Proof. First of all, recall that

$$(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_c})$$
$$(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s})$$
$$\widetilde{\mathbf{e}} = \mathbf{e}_{imp}.$$

We claim

Claim 1. $(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp})$ *is a valid PKE.*

Note that the above claim immediately implies, by definition of $KeyImpose$, that $(\widetilde{\mathbf{g}}, \mathbf{e}_{imp}, \widetilde{\mathbf{d}})$ is also a valid PKE, or equivalently $(\widetilde{\mathbf{g}}, \widetilde{\mathbf{e}}, \widetilde{\mathbf{d}})$ is a valid PKE, as desired. The proof of Claim 1 in turn follows again by a simple inspection of the definition of $EncImpose$.

The proof of Fact 1 is now complete. □

Equipped with Fact 1, toward proving Lemma 11 we bound the probability that $E^{\mathbf{O}}(PK, S; R) \neq E^{\widetilde{\mathbf{O}}}(PK, S; R)$, for a random R. If this probability is small then with high probability $D^{\widetilde{\mathbf{O}}}(SK', C_1 \dots C_n)$ results in S, as desired. We will actually bound a related probability, where S above is replaced with 0^n1^n. (Recall that $|S| = n$.) To this end we need the following lemma.

Lemma 13. *Fix* $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ *and let* $M = 0^n1^n$. *Let* $(qu_1, \dots, qu_{2n^{s+1}})$ *denote the oracle queries asked during the execution of* $E^{\mathbf{O}}(PK, M; R)$, *for a random* R. *Then, for any query index* $1 \leq i \leq 2n^{s+1}$

(A) $$\Pr\left[(qu_i \text{ is } \mathbf{g}\text{- or } \mathbf{e}\text{-type}) \wedge \Big(\forall j < i, \mathbf{O}(qu_j) = \widetilde{\mathbf{O}}(qu_j)\Big) \wedge \Big(\mathbf{O}(qu_i) \neq \widetilde{\mathbf{O}}(qu_i)\Big)\right] \leq \frac{1}{n^{8s}},$$

(B) $$\Pr\left[(qu_i \text{ is } \mathbf{d}\text{-type}) \wedge \Big(\forall j < i, \mathbf{O}(qu_j) = \widetilde{\mathbf{O}}(qu_j)\Big) \wedge \Big(\mathbf{d}(qu_i) \neq \widetilde{\mathbf{d}}(qu_i)\Big)\right] \leq \frac{1}{n^{8s}},$$

where $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \leftarrow Tvars^{\mathcal{O}}(PK)$ *and* R *chosen at random.*

We slightly abused notation above by writing $\widetilde{\mathbf{O}}(qu_j)$, since qu_j is a query to $\mathbf{O}$ (e.g., $qu_j = \langle \mathbf{g}, sk' \rangle$); the meaning, however, should be clear.

We first show how to derive Lemma 11 from Lemma 13.

Proof of Lemma 11. All probabilities that appear below are taken over the choices $(\mathbf{O}, \mathbf{u}, \mathbf{w}, S, PK, C_1, \dots, C_n) \leftarrow Env(n)$ and $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \leftarrow Tvars^{\mathbf{O},\mathbf{u},\mathbf{w}}(PK)$. Let QS be the set of all queries asked during the execution under which $(C_1, \dots, C_n) \leftarrow E^{\mathbf{O}}(PK, S)$ was produced. We claim

$$\Pr[D^{\widetilde{\mathbf{O}}}(SK', C_1 \dots C_n) \neq S] \leq \beta(n) \stackrel{\text{def}}{=} \Pr\left[\exists qu \in \mathsf{QS} \colon \mathbf{O}(qu) \neq \widetilde{\mathbf{O}}(qu)\right].$$

The reason is: if the event inside the right-hand side probability does not hold, then $(C_1, \dots, C_n)$ is also a valid output of $E^{\widetilde{\mathbf{O}}}(PK, S)$. Also, by Fact 1 we know that $(PK, SK') \in G^{\widetilde{\mathbf{O}}}(1^n)$ and that $\widetilde{\mathbf{O}}$ is a correct PKE. Thus, by the correctness of the black-box construction (G, E, D), we obtain $D^{\widetilde{\mathbf{O}}}(SK', C_1 \dots C_n) = S$.

Let QS' contain all queries asked during a random execution of $E^{\mathbf{O}}(PK, M)$, where $M = 0^n 1^n$. We claim

$$\beta(n) \leq \beta'(n) \stackrel{\text{def}}{=} \Pr[\exists qu \in \mathsf{QS}' \colon \mathbf{O}(qu) \neq \widetilde{\mathbf{O}}(qu)]. \tag{4.7.3}$$

Equation 4.7.3 holds because: if S has k 0's, then QS is identically distributed to the set of queries asked during a random execution of $E^{\mathbf{O}}(PK, 0^k 1^{n-k})$. Moreover, since $k \leq n$, the probability that during a random execution of $E^{\mathbf{O}}(PK, 0^k 1^{n-k})$ a query qu, with $\mathbf{O}(qu) \neq \widetilde{\mathbf{O}}(qu)$, is asked is less than the probability that during a random execution of $E^{\mathbf{O}}(PK, M)$ a query qu, with $\mathbf{O}(qu) \neq \widetilde{\mathbf{O}}(qu)$, is asked.

To conclude the proof of Lemma 11 we show $\beta'(n) \leq \frac{1}{2n^5}$. We have

$$\beta'(n) = \Pr\left[\exists qu \in \mathsf{QS}' \colon \mathbf{O}(qu) \neq \widetilde{\mathbf{O}}(qu)\right] \leq 2n^{s+1} \times \frac{1}{n^{8s}} \leq \frac{1}{2n^5};$$

the first inequality is obtained by applying Lemma 13 and a union bound. □

We now prove Lemma 13, starting with Part (A). Toward this goal, we need the following lemma, which characterizes when a query's responses may differ between $\mathbf{g}$ and $\widetilde{\mathbf{g}}$ or between $\mathbf{e}$ and $\widetilde{\mathbf{e}}$.

Lemma 14. *For any* $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ *and* $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \in Tvars^{\mathcal{O}}(PK)$ *all the following hold: (1) for any* $\mathbf{h} \in \{\mathbf{g}, \mathbf{e}\}$ *if* $(\langle \mathfrak{h}, q \rangle, ans) \in \mathsf{Freq}$*, then* $\mathbf{h}(q) = \widetilde{\mathbf{h}}(q) = ans$*; (2) if* $\mathbf{g}(sk) \neq \widetilde{\mathbf{g}}(sk)$ *for some* sk *then* $(\langle \mathfrak{g}, sk \rangle, *) \in \mathsf{Q_s}$*; (3) if* $\mathbf{e}(pk, b, r) \neq \widetilde{\mathbf{e}}(pk, b, r)$ *for some* pk, b *and* r *then either (a)* $(\langle \mathfrak{e}, (pk, b, r) \rangle, *) \in \mathsf{Q_c}$ *or (b) for some* c*:* $(\langle \mathfrak{e}, (pk, *, *) \rangle, c) \in \mathsf{Q_c}$ *and* $\mathbf{u}(pk, c) = (b, r)$.

Proof. First, recall that

$$(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_c})$$
$$(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s})$$
$$\widetilde{\mathbf{e}} = \mathbf{e}_{imp}.$$

We prove all parts of the lemma below.

- (Part 1) We prove the statement for the two types of $\mathfrak{h}$ separately.
  - Suppose $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Freq}$ for some sk and pk. Thus, $\mathbf{g}(sk) = pk$. Moreover, since $\mathsf{Q_s}$ should agree with Freq we have either $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$ or $(\langle \mathfrak{g}, sk\rangle, *) \notin \mathsf{Q_s}$. Both cases imply $\mathbf{g}(sk) = \widetilde{\mathbf{g}}(sk)$.
  - Suppose $(\langle \mathfrak{e}, (pk, b, r)\rangle, c) \in \mathsf{Freq}$, for some pk, b, r and c. Thus, $\mathbf{e}(pk, b, r) = c$. Since $\mathsf{Q_c}$ is generated based on an oracle $(\mathbf{g}', \mathbf{e}', \mathbf{d}')$ that is consistent with Freq and that is also Ψ-valid (in particular, $\mathbf{e}'$ is injective), we have either $(\langle \mathfrak{e}, (pk, b, r)\rangle, c) \in \mathsf{Q_c}$ or $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Q_c}$. Both these conditions imply $\mathbf{e}_{imp}(pk, b, r) = \mathbf{e}(pk, b, r)$. The result now follows by noting that $\widetilde{\mathbf{e}} = \mathbf{e}_{imp}$.
- (Part 2) Straightforward.
- (Part 3) Straightforward by noting that $\widetilde{\mathbf{e}} = \mathbf{e}_{imp}$ and applying the definition of $\mathbf{e}_{imp}$.

□

We also need the following standard theorem. See, e.g., [72] for a proof.

Theorem 17. *(A Chernoff-Hoeffding bound) Let $x_1, \ldots, x_{n^t}$ be independent boolean random variables all identically distributed to x, and suppose $\Pr[x = 1] = p$. Then for $x_{av} = (x_1 + \cdots + x_{n^t})/n^t$*

$$\Pr[|x_{av} - p| \geq \frac{1}{n^k}] \leq \frac{1}{2^{2n^{t-2k}}}. \tag{4.7.4}$$

Proof of Lemma 13, Part (A). Fix $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK)$ as in the lemma and let $\mathsf{QS} = \{qu_1, \ldots, qu_{2n^{s+1}}\}$. All probabilities, if not otherwise stated, are taken over the variables sampled in the lemma. We prove a stronger result, showing

$$\alpha(n) = \Pr\left[(\exists qu \in \mathsf{QS} \text{ such that } qu \text{ is } \mathbf{g}\text{- or } \mathbf{e}\text{-type}) \wedge \left(\mathbf{O}(qu) \neq \widetilde{\mathbf{O}}(qu)\right)\right] \leq \frac{1}{n^{8s}},$$

Let $\mathsf{DifQue_c}$ be formed as follows: for any $(\langle \mathfrak{e}, (pk, b, r)\rangle, c) \in \mathsf{Q_c}$: (a) add $\langle \mathfrak{e}, (pk, b, r)\rangle$ to $\mathsf{DifQue_c}$ and (b) if $\mathbf{u}(pk, c) = (b', r') \neq \perp$, also add $\langle \mathfrak{e}, (pk, b', r')\rangle$ to $\mathsf{DifQue_c}$. By Lemma 14, $\mathsf{DifQue_c}$ is a superset of all $\mathfrak{e}$-type queries that receive different responses from $\mathbf{e}$ and $\widetilde{\mathbf{e}}$. (That is, if $\mathbf{e}(pk, b, r) \neq \widetilde{\mathbf{e}}(pk, b, r)$ then $\langle \mathfrak{e}, (pk, b, r)\rangle \in \mathsf{Q_c}$.) Also, for any $(\langle \mathfrak{g}, sk\rangle, *) \in \mathsf{Q_s}$ add $\langle \mathfrak{g}, sk\rangle$ to the set $\mathsf{DifQue_s}$. Similarly, $\mathsf{DifQue_s}$ is a super-set of all $\mathfrak{g}$-type queries responded to differently under $\mathbf{g}$ and $\widetilde{\mathbf{g}}$. Fix an arbitrary ordering $(\langle \mathfrak{h}_1, q_1\rangle, \ldots, \langle \mathfrak{h}_\mathfrak{m}, q_m\rangle)$ on the elements of $\mathsf{DifQue_c} \cup \mathsf{DifQue_s}$, and note that $m \leq 2n^s$. We claim

$$\alpha(n) \leq \beta(n) \stackrel{\text{def}}{=} \Pr\left[\text{for some } 1 \leq i \leq m\text{: } (\langle \mathbf{h}_i, q_i\rangle \in \mathsf{QS}) \wedge ((\langle \mathfrak{h}_\mathfrak{i}, q_i\rangle, *) \notin \mathsf{Freq})\right].$$

The notation $\langle \mathbf{h}_i, q_i\rangle$ above indicates the corresponding concrete version of the symbolic query $\langle \mathfrak{h}_\mathfrak{i}, q_i\rangle \in \mathsf{DifQue_c} \cup \mathsf{DifQue_s}$. The reason behind the above equation is that if the oracles $\mathbf{O}$ and $\widetilde{\mathbf{O}}$ disagree on a query $\langle \mathbf{h}, q\rangle$, for $\mathbf{h} \in \{\mathbf{g}, \mathbf{e}\}$, then by Lemma 14 $(\langle \mathfrak{h}, q\rangle, *) \notin \mathsf{Freq}$ and $(\langle \mathfrak{h}, q\rangle, *) \in \mathsf{DifQue_c} \cup \mathsf{DifQue_s}$.

For $\mathbf{h} \in \{\mathbf{g}, \mathbf{e}\}$ call $\langle \mathbf{h}, q\rangle$ *heavy* if for some $b \in \{0, 1\}$,

$$\Pr_R\left[\text{the query } \langle \mathbf{h}, q\rangle \text{ is asked during } E^{\mathbf{O}}(PK, b; R)\right] \geq \frac{1}{n^{11s}}. \tag{4.7.5}$$

We call $\langle \mathbf{h}, q\rangle$ *unheavy* if the above probability is strictly less than $\frac{1}{n^{11s}}$ for both $b = 0$ and $b = 1$. Now defining

$$\epsilon(n) = \Pr\left[\exists \text{ query } \langle \mathbf{h}, q\rangle \text{ s.t. } (\langle \mathbf{h}, q\rangle \text{ is heavy }) \wedge (\langle \mathfrak{h}, q\rangle, *) \notin \mathsf{Freq}\right],$$
$$\beta_{1,i}(n) = \Pr\left[(\langle \mathbf{h}_i, q_i\rangle \in \mathsf{QS}) \wedge (\langle \mathfrak{h}_\mathfrak{i}, q_i\rangle \notin \mathsf{Freq}) \wedge (\langle \mathbf{h}_i, q_i\rangle \text{ is unheavy})\right],$$

we have $\beta(n) = \sum_{1 \leq i \leq m}(\epsilon(n) + \beta_{1,i}(n))$. We claim

Claim I. $\beta_{1,i}(n) \leq \frac{1}{2n^{10s}}$, for all $1 \leq i \leq m$.

Claim II. $\epsilon(n)$ is negligible.

Claims I and II imply $\beta(n) \leq \frac{1}{n^{8s}}$, as desired. We now prove the two claims.

Proof of Claim I. Recall that QS contains queries made during $E(PK, 0^n 1^n; R)$ for

a random R. We have

$$\beta_{1,i}(n) \leq \Pr\left[(\langle \mathbf{h}_i, q_i\rangle \in \mathsf{QS}) \wedge (\langle \mathbf{h}_i, q_i\rangle \text{ is unheavy})\right] \leq \Pr\left[\langle \mathbf{h}_i, q_i\rangle \in \mathsf{QS} \mid \langle \mathbf{h}_i, q_i\rangle \text{ is unheavy}\right]$$

and thus $\beta_{1,i}(n) \leq 2n/(n^{11s}) \leq 1/(n^{9s})$, as claimed. Note we crucially used the fact R is independent of $\langle \mathbf{h}_i, q_i\rangle$: the latter depends only on $\mathsf{Q_s} \cup \mathsf{Q_c}$ and $\mathbf{O}$, while R is chosen independently of $\mathbf{O}$ and $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \leftarrow Tvars^{\mathcal{O}}(PK)$.

Proof of Claim II. The proof consists of two parts: first, we show there are at most a poly number of heavy queries (**Claim A**); next, we show for a specific heavy query $\langle \mathbf{h}, q\rangle$ the probability that $(\langle \mathfrak{h}, q\rangle, *) \notin \mathsf{Freq}$ is negligible (**Claim B**).

Proof of Claim A. Fix $b \in \{0,1\}$. We show the number of queries $\langle \mathbf{h}, q\rangle$ for which Equation 4.7.5 holds for the fixed b is at most polynomial. Call $\langle \mathbf{h}, q\rangle$ *i-heavy* if with probability at least $1/(n^{12s})$ $\langle \mathbf{h}, q\rangle$ is the ith query during $E^{\mathbf{O}}(PK, b)$. If $\langle \mathbf{h}, q\rangle$ is asked with probability at least $1/(n^{11s})$ during $E^{\mathbf{O}}(PK, b)$ then for some $1 \leq i \leq n^s$ $\langle \mathbf{h}, q\rangle$ is i-heavy. For every i we have at most n^{12s} i-heavy queries. Thus, for a fixed b we have at most $n^{12s} \times n^s$ queries for which Equation 4.7.5 holds. Thus, we have at most $2n^{13s}$ heavy queries.

Proof of Claim B. Recall that Freq is formed by collecting all query/response pairs made during n^{23s} random executions of $E^{\mathbf{O}}(PK, 01)$. For $1 \leq k \leq n^{23s}$ let $x_k = 1$ if $\langle \mathbf{h}, q\rangle$ is asked during the kth execution, and $x_k = 0$, otherwise. We know that $x_1, \ldots, x_{n^{23s}}$ are all identically distributed and independent, and that $p \stackrel{\text{def}}{=} \Pr[x_1 = 1] \geq 1/(n^{11s})$. Let $x_{av} = (x_1 + \cdots + x_{n^{23s}})/n^{23s}$. We have

$$\Pr[(\langle \mathfrak{h}, q\rangle, *) \notin \mathsf{Freq} \mid \langle \mathbf{h}, q\rangle \text{ is heavy}] \leq \Pr[|x_{av} - p| \geq p] \leq \Pr[|x_{av} - p| \geq \frac{1}{n^{11s}}].$$

By Theorem 17 the last probability above is at most $\frac{1}{2^{2n^{23s-22s}}} \leq \frac{1}{2^{2n}}$. □

We now prove Part (B) of Lemma 13. Similarly to the previous part we first characterize when for an arbitrary (sk, c) we may have $\mathbf{d}(sk, c) \neq \widetilde{\mathbf{d}}(sk, c)$.

Lemma 15. *Let* $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ *and* $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \in Tvars^{\mathcal{O}}(PK)$. *For* (sk, c) *suppose* $\widetilde{\mathbf{g}}(sk) = \mathbf{g}(sk) = pk$, *but* $\mathbf{d}(sk, c) \neq \widetilde{\mathbf{d}}(sk, c)$. *Then all the following conditions hold: (1)* $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \in \mathsf{Q_c}$; *(2)* $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Freq}$; *and (3)* $(\langle \mathfrak{d}, (sk, c)\rangle, *) \notin \mathsf{Freq}$.

Proof. Suppose $\widetilde{\mathbf{g}}(sk) = \mathbf{g}(sk) = pk$ and $\mathbf{d}(sk, c) \neq \widetilde{\mathbf{d}}(sk, c)$. Recall that

$$(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_c})$$
$$(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s})$$
$$\widetilde{\mathbf{e}} = \mathbf{e}_{imp}.$$

To prove the lemma we first prove the following claim.

Claim 2. *Assuming* $\widetilde{\mathbf{g}}(sk) = \mathbf{g}(sk) = pk$ *it holds that* $\widetilde{\mathbf{d}}(sk, c) = \mathbf{d}_{imp}(sk, c)$

Proof of Claim 2. Recall by Claim 1 that $(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp})$ is a valid PKE. For $b \in \{0, 1\}$ have

$$\widetilde{\mathbf{d}}(sk, c) = b \Leftrightarrow \mathbf{e}_{imp}(pk, b, *) = c \Leftrightarrow \mathbf{d}_{imp}(sk, c) = b.$$

Also, we have

$$\widetilde{\mathbf{d}}(sk, c) = \perp \Leftrightarrow \text{for no } b\text{: } \mathbf{e}_{imp}(pk, b, *) = c \Leftrightarrow \mathbf{d}_{imp}(sk, c) = \perp.$$

Thus, we have $\widetilde{\mathbf{d}}(sk, c) = \mathbf{d}_{imp}(sk, c)$.

We now show that the negation of any of the conditions claimed in the lemma implies $\mathbf{d}(sk, c) = \mathbf{d}_{imp}(sk, c)$, which in turn by Claim 2 implies $\mathbf{d}(sk, c) = \widetilde{\mathbf{d}}(sk, c)$, which is a contradiction to the assumption of the lemma that $\mathbf{d}(sk, c) \neq \widetilde{\mathbf{d}}(sk, c)$.
$\neg(1)$ **holds:** Then immediately from the definition of $\mathbf{d}_{imp}$ we obtain that $\mathbf{d}(sk, c) = \mathbf{d}_{imp}(sk, c)$.
$\neg(2)$ **holds:** Then we have $(\langle \mathfrak{e}, (pk, b, *)\rangle, c) \in \mathsf{Freq}$, for some b. Thus, $\mathbf{d}(sk, c) = b$. Now in order for $\mathbf{d}(sk, c) \neq \mathbf{d}_{imp}(sk, c)$ to be true, it must hold that $(\langle \mathfrak{e}, (pk, 1 - b, *)\rangle, c) \in \mathsf{Q_c}$, which is impossible since $\mathsf{Q_c}$ agrees with Freq.
$\neg(3)$ **holds:** Thus, $(\langle \mathfrak{d}, (sk, c)\rangle, *) \in \mathsf{Freq}$. That is, either $(\langle \mathfrak{d}, (sk, c)\rangle, \perp) \in \mathsf{Freq}$ or $(\langle \mathfrak{d}, (sk, c)\rangle, b) \in \mathsf{Freq}$ for some bit b. We consider both cases:

- If $(\langle \mathfrak{d}, (sk, c)\rangle, \perp) \in \mathsf{Freq}$, then by Assumption 3 we have $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Freq}$. Now since $\mathsf{Q_c}$ is obtained from some $\mathbf{e}'$, where $(\mathbf{g}', \mathbf{e}', \mathbf{d}')$ is a valid PKE that agrees with Freq (see Definition 21) we have $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Q_c}$. (To see this note that $\mathbf{d}'(sk, c) = \perp$ and that $\mathbf{g}'(sk) = pk$, since $(\langle \mathfrak{d}, (sk, c)\rangle, \perp) \in \mathsf{Freq}$ and $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Freq}$. If $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \in \mathsf{Q_c}$ then $\mathbf{e}'(pk, *, *) = c$, and since $\mathbf{g}'(sk) = pk$, it holds that $\mathbf{d}'(sk, c) \neq \perp$, which is a contradiction to the earlier

established fact that $\mathbf{d}'(sk, c) = \perp$.) Now the fact that $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Q_c}$ implies $\mathbf{d}(sk, c) = \mathbf{d}_{imp}(sk, c)$.

- If $(\langle \mathfrak{d}, (sk, c)\rangle, b) \in \mathsf{Freq}$ for some bit b, then by the way in which Freq is sampled we have $(\langle \mathfrak{e}, (pk, b, *)\rangle, c) \in \mathsf{Freq}$. Now since $\mathsf{Q_c}$ should agree with Freq it holds either that (a) $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Q_c}$ or (b) if $(\langle \mathfrak{e}, (pk, b', *)\rangle, c) \in \mathsf{Q_c}$ then $b' = b$. Both (a) and (b) imply $\mathbf{d}_{imp}(sk, c) = b$, and thus $\mathbf{d}(sk, c) = \mathbf{d}_{imp}(sk, c)$.

The proof of Lemma 15 is now complete. □

Lemma 13, Part (B) (Restated). *Fix $(\mathbf{O}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ and let $M = 0^n 1^n$. Let $(qu_1, \ldots, qu_{2n^{s+1}})$ denote the oracle queries asked during the execution of $E^{\mathbf{O}}(PK, M; R)$. Then, for any query index $1 \le i \le 2n^{s+1}$*

$$\Pr\left[(qu_i \textit{ is } \mathbf{d}\textit{-type}) \wedge \Big(\forall j < i, \mathbf{O}(qu_j) = \widetilde{\mathbf{O}}(qu_j)\Big) \wedge \Big(\mathbf{d}(qu_i) \neq \widetilde{\mathbf{d}}(qu_i)\Big)\right] \le \frac{1}{n^{8s}},$$

where $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}}) \leftarrow Tvars^{\mathcal{O}}(PK)$ and R chosen at random.

Proof. Fix $i \in \{1, \ldots, 2n^{s+1}\}$ as in the lemma. All probabilities below are taken over the random coins described in the lemma, unless otherwise stated. Define the random variable $(\mathit{sk}, \mathit{c})$ as follows: $(\mathit{sk}, \mathit{c}) = (sk_i, c_i)$ if $qu_i = \langle \mathbf{d}, (sk_i, c_i)\rangle$, and $(\mathit{sk}, \mathit{c}) = (\perp, \perp)$, otherwise. We show $\beta(n) \le \frac{1}{n^{10s}}$, where

$$\Pr[\underbrace{((\mathit{sk}, \mathit{c}) \neq (\perp, \perp))}_{Evnt1} \wedge \underbrace{(\mathbf{d}(\mathit{sk}, \mathit{c}) \neq \widetilde{\mathbf{d}}(\mathit{sk}, \mathit{c}))}_{Evnt2} \wedge \underbrace{(\forall j < i, \mathbf{O}(qu_j) = \widetilde{\mathbf{O}}(qu_j))}_{Evnt3}] \le \frac{1}{n^{10s}}.$$

Define pk to be $\mathbf{g}(\mathit{sk})$ if $\mathit{sk} \neq \perp$, and $\mathit{pk} = \perp$, otherwise. If we have that $Evnt1 \wedge Evnt2 \wedge Evnt3$ holds, since qu_i is the first query where $\mathbf{O}(qu_i) \neq \widetilde{\mathbf{O}}(qu_i)$, by Assumption 3 we have $\widetilde{\mathbf{g}}(\mathit{sk}) = \mathit{pk} \neq \perp$. Moreover, by Lemma 15 we have $(\langle \mathfrak{e}, (\mathit{pk}, *, *)\rangle, c) \in \mathsf{Q_c}$, $(\langle \mathfrak{e}, (\mathit{pk}, *, *)\rangle, c) \notin \mathsf{Freq}$ and $(\langle \mathfrak{d}, (\mathit{sk}, c)\rangle, *) \notin \mathsf{Freq}$.

Fix an arbitrary ordering $((\langle \mathfrak{e}, (pk_1, b_1, r_1)\rangle, c_1), \ldots, (\langle \mathfrak{e}, (pk_m, b_m, r_m)\rangle, c_m))$ on the elements of $\mathsf{Q_c}$. Note that $m \le n^s$. By the discussion above if $Evnt1 \wedge Evnt2 \wedge Evnt3$ holds, then the following must hold:

- for some $h \le m$, $(\mathit{pk}, \mathit{c}) = (pk_h, c_h)$, $(\langle \mathfrak{e}, (\mathit{pk}, *, *)\rangle, c) \notin \mathsf{Freq}$ and $(\langle \mathfrak{d}, (\mathit{sk}, \mathit{c})\rangle, *) \notin \mathsf{Freq}$.

Thus, we have $\beta(n) \le \sum_{h=1}^{m} \beta_h(n)$, where

$$\beta_h(n) = \Pr\left[\Big((\mathit{pk}, \mathit{c}) = (pk_h, c_h)\Big) \wedge \Big(\langle \mathfrak{e}, (\mathit{pk}, *, *)\rangle, c) \notin \mathsf{Freq}\Big) \wedge (\langle \mathfrak{d}, (\mathit{sk}, \mathit{c})\rangle, *) \notin \mathsf{Freq})\right].$$

We now claim

Claim I. For all $1 \leq h \leq m$, $\beta_h(n) \leq \frac{1}{n^{10s}}$.

Claim I implies the desired bound: $\beta(n) \leq m \times \frac{1}{n^{10s}} \leq \frac{1}{n^{8s}}$. Thus, we focus on proving Claim I. Fix $h \in \{1, \ldots, m\}$ in the sequel.

Call a pair (pk, c), with $sk = \mathbf{g}^{-1}(pk)$, *heavy* if for some $b \in \{0, 1\}$,

$$\Pr_R\left[(\langle \mathbf{e}, (pk, *, *)\rangle, c) \text{ or } (\langle \mathbf{d}, (sk, c)\rangle, *) \text{ is made during } E^{\mathbf{O}}(PK, b; R)\right] \geq \frac{1}{n^{11s}}. \tag{4.7.6}$$

Note that in the equation above it may be that $sk = \perp$, in which case the query $\langle \mathbf{d}, (sk, c)\rangle$ is invalid, and so it is never asked, and thus the above probability amounts to the probability of the former condition alone. We call (pk, c) *unheavy* if the above probability is strictly less than $\frac{1}{n^{11s}}$ for both $b = 0$ and $b = 1$.

Recalling h fixed above, letting $sk_h = \mathbf{g}^{-1}(pk_h)$, to bound $\beta_h(n)$ we define the events Ev_1 and Ev_2:

1. Ev_1: there exists (pk, c) such that (pk, c) is heavy and $(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Freq}$ and $(\langle \mathfrak{d}, (\mathbf{g}^{-1}(pk), c)\rangle, *) \notin \mathsf{Freq}$

2. Ev_2: $(\mathcal{pk}, \mathcal{c}) = (pk_h, c_h)$, $(\langle \mathbf{e}, (\mathcal{pk}, *, *)\rangle, \mathcal{c}) \notin \mathsf{Freq}$, $(\langle \mathbf{d}, (\mathcal{sk}, \mathcal{c})\rangle, *) \notin \mathsf{Freq}$ and (pk_h, c_h) is unheavy.

Letting $\epsilon(n) = \Pr[Ev_1]$ and $\beta_{h,1}(n) = \Pr[Ev_2]$ we have $\beta_h(n) \leq \epsilon(n) + \beta_{h,1}(n)$. We show $\epsilon(n) \leq \frac{1}{n^{11s}}$ and $\beta_{h,1}(n) \leq \frac{1}{n^{11s}}$, and this implies Claim A.

Bounding $\epsilon(n)$. The proof consists of two parts: first, we show there are at most a poly number of heavy pairs (pk, c) (**Claim A**); next, we show for a specific heavy pair (pk, c) the probability that $(\langle \mathfrak{h}, q\rangle, *) \notin \mathsf{Freq}$ is negligible (**Claim B**).

Proof of Claim A. Fix $b \in \{0, 1\}$. We show the number of pairs (pk, c) for which Equation 4.7.6 holds for the fixed b is at most polynomial. Call (pk, c), with $sk = \mathbf{g}^{-1}(pk)$, *i-heavy* if during a random execution of $E^{\mathbf{O}}(PK, b; R)$ with probability at least $\frac{1}{n^{12s}}$ (taken over R) the ith query/response pair is either $(\langle \mathbf{e}, (pk, *, *)\rangle, c)$ or $(\langle \mathbf{d}, (sk, c)\rangle, *)$. If at least one of $(\langle \mathbf{e}, (pk, *, *)\rangle, c)$ or $(\langle \mathbf{d}, (sk, c)\rangle, *)$ is asked with probability at least $\frac{1}{n^{11s}}$ during $E^{\mathbf{O}}(PK, b; R)$ then for some $1 \leq i \leq n^s$ (pk, c) is i-heavy. For every i we have at most n^{12s} i-heavy pairs. Thus, for a fixed b we have at most $n^{12s} \times n^s$ pairs for which Equation 4.7.6 holds. Thus, we have at most $2n^{13s}$ heavy pairs (pk, c), and the proof of Claim A is complete.

Proof of Claim B. Recall that Freq is formed by collecting all query/response pairs made during n^{23s} random executions of $E^{\mathbf{O}}(PK, 01)$. For $1 \leq k \leq n^{23s}$ let $x_k = 1$ if at least one of $(\langle \mathbf{e}, (pk, *, *)\rangle, c)$ and $(\langle \mathbf{d}, (sk, c)\rangle, *)$ is asked during the kth execution, and $x_k = 0$, otherwise. We know that $x_1, \ldots, x_{n^{23s}}$ are all identically distributed and independent, and that $p \stackrel{\text{def}}{=} \Pr[x_1 = 1] \geq \frac{1}{n^{11s}}$. Let $x_{av} = \frac{x_1 + \cdots + x_{n^{23s}}}{n^{23s}}$. We have

$$\begin{aligned}&\Pr[(\langle \mathfrak{e}, (pk, *, *)\rangle, c) \notin \mathsf{Freq}) \wedge (\langle \mathfrak{d}, (sk, c)\rangle, *) \notin \mathsf{Freq}) \mid (pk, c) \text{ is heavy}] \\ &\leq \Pr[|x_{av} - p| \geq p] \leq \Pr[|x_{av} - p| \geq \frac{1}{n^{11s}}] \leq \frac{1}{2^{2n^{23s-22s}}} \leq \frac{1}{2^{2n}},\end{aligned}$$

where the third inequality above follows by Theorem 17. The proof of Claim B is complete.

Bounding $\beta_{h,1}(n)$.

$$\begin{aligned}\beta_{h,1}(n) &\leq \Pr\left[\left((\mathit{pk}, \mathit{c}) = (pk_h, c_h)\right) \wedge ((pk_h, c_h) \text{ is unheavy})\right] \\ &\leq \Pr\left[(\mathit{pk}, \mathit{c}) = (pk_h, c_h) \mid (pk_h, c_h) \text{ is unheavy}\right] \leq \frac{1}{n^{11s}}.\end{aligned}$$

The reason for the last inequality above is that since the index i, to which $(\mathit{pk}, \mathit{c})$ corresponds, is fixed, the probability that $(\mathit{pk}, \mathit{c}) = (pk_h, c_h)$ is at most the probability that at least one of $(\langle \mathbf{e}, (pk_h, *, *)\rangle, c_h)$ or $(\langle \mathbf{d}, (sk_h, c_h)\rangle, *)$ is made during a random encryption $E^{\mathbf{O}}(PK, b)$: Here b is the bit of $0^n 1^n$ to which the index i corresponds, and b is uniquely determined by Assumption 2.

The proof of Part (B) of Lemma 13 is now complete. □

4.7.2 Proof of Lemma 12

The following proof generalizes a similar proof sketched in Section 4.5.

Proof. Let $\alpha(n)$ be as in the lemma. Also, let $\mathbf{Env} = (\mathbf{O}, \mathbf{u}, \mathbf{w}, S, PK, C_1, \ldots, C_n)$ and $\mathbf{T}$ be sampled at random and let Freq, SK', $\mathsf{Q_s}$ and $\widetilde{\mathbf{O}}$ be the variables formed during the execution of $\mathbf{T}(PK, C_1, \ldots, C_n)$. We first introduce some notation.

- $FreqPub(\mathsf{Freq}) = \{pk \mid (\langle \mathfrak{g}, *\rangle, pk) \in \mathsf{Freq}\}$.
- Let $QPub^{\mathbf{w}}(\mathsf{Q_s})$: the set of pk's such that (a) $|pk| \geq 5 \log n^{25s}$, (b) $\mathbf{w}(pk) = \top$ and (c) $(\langle \mathfrak{g}, *\rangle, pk) \in \mathsf{Q_s}$. (We put $\mathbf{w}$ as a superscript to $QPub$ to emphasize that the formation of this set involves making $\mathbf{w}$ queries.)

- $EmbedPub^{\mathbf{O}}(S) = \{pk \mid$ a query/response $(\langle \mathbf{g}, * \rangle, pk)$ is made during $G^{\mathbf{O}}(S)\}$.

To bound $\alpha(n)$, suppose $\mathbf{T}(PK, C_1, \ldots, C_n) = \perp$ and $D^{\tilde{\mathbf{O}}}(SK', C_1 \ldots C_n) = S$. Then by Step 5 of $\mathbf{T}$'s computation it must hold that

$$QPub^{\mathbf{w}}(\mathsf{Q_s}) \not\subseteq EmbedPub^{\mathbf{O}}(S) \cup FreqPub(\mathsf{Freq}).$$

Thus,

$$\alpha(n) \leq \Pr_{\mathbf{Env},\mathbf{T}}[QPub^{\mathbf{w}}(\mathsf{Q_s}) \not\subseteq EmbedPub^{\mathbf{O}}(S) \cup FreqPub(\mathsf{Freq})]. \tag{4.7.7}$$

We build an adversary $\mathcal{B}^{\mathcal{g},\mathcal{e},\mathcal{d},u,w}(1^n)$ that makes a set Que of at most $P = n^{25s}$ queries and outputs a public key pk, s.t. $|pk| \geq 5 \log P$, satisfying

$$\Pr_{(\mathcal{g},\mathcal{e},\mathcal{d},u,w) \leftarrow \Psi} \left[w(pk) = \top \wedge (\langle \mathcal{g}, * \rangle, pk) \notin \mathsf{Que} \right] \geq \alpha(n), \tag{4.7.8}$$

Then from Lemma 9, we conclude $\alpha(n) \leq \frac{1}{n^{25s}} \leq \frac{1}{n^{25}}$.
Let $\mathcal{O} = (\mathcal{g}, \mathcal{e}, \mathcal{d})$. The adversary $\mathcal{B}^{\mathcal{O},u,w}(1^n)$ works as follows:

1. $\mathcal{B}$ samples $\mathcal{S} \leftarrow \{0,1\}^n$ and runs $G^{\mathcal{O}}(\mathcal{S})$ to get $(\mathcal{PK}, \mathcal{SK})$, and for any pair of query/response $(\langle \mathcal{g}, * \rangle, pk)$ made, it adds pk to $\mathcal{EmbedPub}$. Note $\mathcal{EmbedPub} = EmbedPub^{\mathcal{O}}(\mathcal{S})$.

2. $\mathcal{B}$ samples $\mathcal{Freq} \leftarrow FreqQue^{\mathcal{O},u}(\mathcal{PK}, n^{23s})$ and then samples $(\mathcal{SK}', \mathcal{Q_s}, \mathcal{Q_c})$ by running $ConsOrc(\mathcal{PK}, \mathcal{Freq})$.

3. $\mathcal{B}$ forms $\mathcal{FreqPub} = FreqPub(\mathcal{Freq})$ and $\mathcal{QPub} = QPub^{w}(\mathcal{Q_s})$. If there is $pk \in \mathcal{QPub} \setminus (\mathcal{EmbedPub} \cup \mathcal{FreqPub})$, $\mathcal{B}$ returns pk; else it returns $pk \leftarrow \{0,1\}^{5 \log P}$.

We bound $|\mathsf{Que}|$: $\mathcal{B}$ makes n^s queries to produce $(\mathcal{PK}, \mathcal{SK})$, at most $4n^{24s}$ queries to sample $\mathcal{Freq}$ (see Definition 20), and at most n^s queries to produce $\mathcal{QPub}$. The rest are computed offline. Thus, $|\mathsf{Que}| \leq 4n^{24s} + 2n^s < n^{25s}$.

To analyze $\mathcal{B}$'s success probability, note $(\mathbf{O}, \mathbf{u}, \mathbf{w}, S, \mathsf{Freq}, \mathsf{Q_s})$, sampled as in Equation 4.7.7, is identically distributed to $(\mathcal{O}, u, w, \mathcal{S}, \mathcal{Freq}, \mathcal{Q_s})$. Also, note for any pk it holds that $pk \in \mathcal{EmbedPub} \cup \mathcal{FreqPub}$ iff $(\langle \mathcal{g}, * \rangle, pk) \in \mathsf{Que}$. Also, by definition if $pk \in \mathcal{QPub}$ then $|pk| \geq 5 \log P$ and $w(pk) = \top$. Thus, from Equation 4.7.7, we deduce $\mathcal{B}$'s success probability is greater than $\alpha(n)$, completing the proof. $\square$

4.8 T Does Not Break the CPA Security of $(\mathbf{g}, \mathbf{e}, \mathbf{d})$

We prove the stronger result that $\mathbf{T}$ does not break the CCA2 security of $(\mathbf{g}, \mathbf{e}, \mathbf{d})$.

Theorem 18. *Suppose $\mathcal{A}$ is a CCA2-valid adversary with access to oracles $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w})$ and $\mathbf{T}$ that makes at most $2^{n/8}$ queries. We have*

$$\Pr_{\mathcal{O},\mathbf{T},b,pk,c}[\mathcal{A}^{\mathcal{O},\mathbf{T}}(1^n, pk, c) = b] \leq \frac{1}{2} + \frac{1}{2^{n/4}}, \tag{4.8.1}$$

where $\mathcal{O} \leftarrow \Psi$, $b \leftarrow \{0,1\}$, $sk \leftarrow \{0,1\}^n$, $pk = \mathbf{g}(sk)$ and $c \leftarrow \mathbf{e}(pk, b)$.

The following lemma is used in the proof of Theorem 18: it shows how to simulate responses to queries to $\widetilde{\mathbf{O}}$ via oracle access to $(\mathbf{O}, \mathbf{u}, \mathbf{w})$.

Lemma 16. *Fix $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ and also fix*

$$(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{g}}, \widetilde{\mathbf{e}}, \widetilde{\mathbf{d}}) \in Tvars^{\mathcal{O}}(PK).$$

Assuming $\mathsf{Q_c} = \{(\langle \mathfrak{e}, (pk_1, b_1, r_1)\rangle, c_1), \ldots, (\langle \mathfrak{e}, (pk_p, b_p, r_p)\rangle, c_p)\}$ let $\mathsf{W} = \{(pk_i, c_i) \colon 1 \leq i \leq p\}$ and $\mathsf{W}' = \{(pk_i, \mathbf{e}(pk_i, b_i, r_i)) \colon 1 \leq i \leq p\}$. Then

(a) Both $\widetilde{\mathbf{g}}$ and $\widetilde{\mathbf{e}}$ can be computed efficiently (on all points) given access to oracles $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$ and having $\mathsf{Q_s}$ and $\mathsf{Q_c}$ as input.

(b) For any (sk, c), if $(\langle \mathfrak{g}, sk\rangle, *) \notin \mathsf{Q_s}$, the value $\widetilde{\mathbf{d}}(sk, c)$ can be efficiently computed given access to oracle $\mathbf{O}$ and having $\mathsf{Q_c}$ as input.

(c) If for (sk, c) it holds that $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$ for some pk and that $(pk, c) \notin \mathsf{W} \cup \mathsf{W}'$, then $\widetilde{\mathbf{d}}(sk, c)$ can be determined as follows: if $\mathbf{u}(pk, c) = (b, *) \neq \perp$ then $\widetilde{\mathbf{d}}(sk, c) = b$; otherwise, $\widetilde{\mathbf{d}}(sk, c) = \perp$.

(d) If for (sk, c) it holds that $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$, for some pk, and that $(pk, c) \in \mathsf{W}' \setminus \mathsf{W}$, then $\widetilde{\mathbf{d}}(sk, c) = \perp$.

(e) If for (sk, c) it holds that $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$, for some pk, and that $(pk, c) = (pk_i, c_i)$ for some $i \leq p$, then $\widetilde{\mathbf{d}}(sk, c) = b_i$.

Proof. The proof follows mostly by applying definitions. First of all, recall that

$$(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathsf{Q_c})$$
$$(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s})$$
$$\widetilde{\mathbf{e}} = \mathbf{e}_{imp}.$$

Part (a). The proof of this case follows easily by inspection. We give the proof for $\widetilde{\mathbf{e}}$, since this is arguably the harder case. To compute $\widetilde{\mathbf{e}}(pk, b, r)$: if $(\langle \mathfrak{e}, (pk, b, r)\rangle, c) \in \mathsf{Q_c}$ for some c, then $\widetilde{\mathbf{e}}(pk, b, r) = c$; else if $(pk, \mathbf{e}(pk, b, r)) \notin \mathsf{W}$ then $\widetilde{\mathbf{e}}(pk, b, r) = \mathbf{e}(pk, b, r)$; else $\widetilde{\mathbf{e}}(pk, b, r) = \mathbf{e}(pk, b, r + x)$ where x is the smallest integer such that $(pk, \mathbf{e}(pk, b, r + x)) \notin \mathsf{W} \cup \mathsf{W}'$.

Part (b). If $(\langle \mathfrak{g}, sk\rangle, *) \notin \mathsf{Q_s}$ then by Claim 2 we have $\widetilde{\mathbf{d}}(sk, c) = \mathbf{d}_{imp}(sk, c)$. To compute $\mathbf{d}_{imp}(sk, c)$, first compute $pk = \mathbf{g}(sk)$; if $(\langle \mathfrak{e}, (pk, b, *)\rangle, c) \in \mathsf{Q_c}$ for some bit b, then $\mathbf{d}_{imp}(sk, c) = b$; otherwise, by definition $\mathbf{d}_{imp}(sk, c) = \mathbf{d}(sk, c)$.

Part (c). If for some pk, $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$, then by definition $\widetilde{\mathbf{d}}(sk, c)$ is computed as follows: if for some b and r it holds that $c = \mathbf{e}_{imp}(pk, b, r)$ then $\widetilde{\mathbf{d}}(sk, c) = b$; otherwise, $\widetilde{\mathbf{d}}(sk, c) = \perp$. Now suppose $(pk, c) \notin \mathsf{W} \cup \mathsf{W}'$ (as it is in this part of the lemma). If $\mathbf{u}(pk, c) = \perp$ we claim that for no b and r it holds that $c = \mathbf{e}_{imp}(pk, b, r)$; the reason is that since $(pk, c) \notin \mathsf{W}$ if $c = \mathbf{e}_{imp}(pk, b, r)$ then for some r' it holds $\mathbf{e}(pk, b, r') = c$ (this can be easily verified from the definition of $\mathbf{e}_{imp}$), which contradicts $\mathbf{u}(pk, c) = \perp$. Thus, $\widetilde{\mathbf{d}}(sk, c) = \perp$. On the other hand, if $\mathbf{u}(pk, c) = (b, *) \neq \perp$, then we know for some r we have $\mathbf{e}(pk, b, r) = c$. We claim $\mathbf{e}_{imp}(pk, b, r) = c$. Assume not: then by definition for some $i \leq p$ it holds either $(pk, b, r) = (pk_i, b_i, r_i)$ or $(pk, \mathbf{e}(pk, b, r)) = (pk_i, c_i)$. The former implies $(pk, c) \in \mathsf{W}'$ and the latter implies $(pk, c) \in \mathsf{W}$, both contradicting the assumption $(pk, c) \notin \mathsf{W} \cup \mathsf{W}'$.

Part (d). If for some pk, $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$, then, again by definition, $\widetilde{\mathbf{d}}(sk, c) = b$ iff there exists b and r such that $c = \mathbf{e}_{imp}(pk, b, r)$. Since $(pk, c) \in \mathsf{W}'$ we know $pk = pk_i$ and $c = \mathbf{e}(pk_i, b_i, r_i)$ for some $i \leq p$. Assume toward a contradiction that for some b and r it holds $c = \mathbf{e}_{imp}(pk, b, r)$. Thus, $\mathbf{e}_{imp}(pk, b, r) = c = \mathbf{e}(pk_i, b_i, r_i)$. We consider all possible cases, and prove each yields a contradiction:

- $(pk, b, r) = (pk_j, b_j, r_j)$ for some $j \leq p$: impossible since otherwise $\mathbf{e}_{imp}(pk, b, r) = c_j$ and so $(pk, c) = (pk_j, c_j) \in \mathsf{W}$.
- $(pk, \mathbf{e}(pk, b, r)) \notin \mathsf{W}'$: then since by Part (a) we know for no $j \leq p$ $(pk, b, c) = (pk_j, b_j, c_j)$, by applying the definition of $\mathbf{e}_{imp}$ we obtain $\mathbf{e}_{imp}(pk, b, r) = \mathbf{e}(pk, b, r)$.

Since from earlier we have $\mathbf{e}_{imp}(pk, b, r) = \mathbf{e}(pk_i, b_i, r_i)$ and $pk = pk_i$, we obtain $(pk, \mathbf{e}(pk, b, r)) = (pk_i, \mathbf{e}(pk_i, b_i, r_i)) \in \mathsf{W}'$, which is a contradiction to the assumption that $(pk, \mathbf{e}(pk, b, r)) \notin \mathsf{W}'$.

- $(pk, \mathbf{e}(pk, b, r)) \in \mathsf{W}'$: Since by Part (a) we know for no $h \leq p$ it holds that $(pk, b, c) = (pk_h, b_h, c_h)$, by applying the definition of $\mathbf{e}_{imp}$ we obtain $\mathbf{e}_{imp}(pk, b, r) = \mathbf{e}(pk, b, r + x)$ for some x for which in particular $(pk, b, r + x) \neq (pk_i, b_i, r_i)$. Since from earlier we have $pk = pk_i$, we obtain $(b, r + x) \neq (b_i, r_i)$. Thus, we have $\mathbf{e}(pk, b, r + x) = c = \mathbf{e}(pk, b_i, r_i)$, which is a contradiction since $\mathbf{e}(pk, \cdot, \cdot)$ is one-to-one.

Part (e). If for some pk, $(\langle \mathfrak{g}, sk\rangle, pk) \in \mathsf{Q_s}$, then by definition $\widetilde{\mathbf{d}}(sk, c)$ is computed as follows: if there exists b and r such that $\mathbf{e}_{imp}(pk, b, r) = c$ then $\widetilde{\mathbf{d}}(sk, c) = b$; otherwise, $\widetilde{\mathbf{d}}(sk, c) = \perp$. Now since $(pk, c) = (pk_i, c_i)$, we have $\mathbf{e}_{imp}(pk, b_i, r_i) = c$ and so $\widetilde{\mathbf{d}}(sk, c) = b_i$.

The proof of Lemma 16 is now complete. □

We first give a sketch of the proof of Theorem 18 and then present the full proof.

Proof sketch of Theorem 18 As in Section 4.5 the idea is to give an adversary $\mathcal{B}$, where $\mathcal{B}^{\mathcal{O}}(1^n, pk, c)$ can simulate responses to $\mathbf{T}$ queries of $\mathcal{A}^{\mathcal{O},\mathbf{T}}(1^n, pk, c)$, without calling $\langle \mathbf{u}, (pk, c)\rangle$. Let $Tqu = \langle \mathbf{T}, (1^n, PK, C_1, \ldots, C_n)\rangle$ be an $\mathcal{A}$'s query. As per $\mathbf{T}$'s computation, $\mathcal{B}$ first samples $\mathsf{Freq} \leftarrow FreqQue^{\mathbf{O},\mathbf{u}}(PK, n^{23s})$. This may seem problematic since this step involves making $\mathbf{u}$ queries. By inspecting Definition 20, however, we can see that for any query $\langle \mathbf{u}, (pk', *)\rangle$ that needs to be made, $\mathcal{B}$ already knows $\mathbf{g}^{-1}(pk')$. Finally, let $\mathsf{FreqPub}$ contain any pk' such that $(\langle \mathfrak{g}, *\rangle, pk') \in \mathsf{Freq}$, and assume w.l.o.g. $pk \notin \mathsf{FreqPub}$ (because otherwise $\mathcal{B}$ has already found $\mathbf{g}^{-1}(pk)$.)

Next, $\mathcal{B}$ samples $(SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{O}})$ as in $\mathbf{T}$'s execution and starts simulating the decryption $D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots, C_n)$. Again the idea is to see if $(\langle \mathfrak{g}, *\rangle, pk) \in \mathsf{Q_s}$ or not. If $(\langle \mathfrak{g}, *\rangle, pk) \notin \mathsf{Q_s}$: by Lemma 16 we can see $\mathcal{B}$ can handle all $\widetilde{\mathbf{O}}$ queries. In particular, $\mathcal{B}$ will never need to call $\langle \mathbf{u}, (pk, *)\rangle$. After the decryption $\mathcal{B}$ performs Step 5 of $\mathbf{T}$'s computation, which $\mathcal{B}$ can efficiently do, since no $\mathbf{u}$ queries are involved.

If, however, $(\langle \mathfrak{g}, *\rangle, pk) \in \mathsf{Q_s}$, assuming $S_{out} = D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots, C_n)$, since $pk \notin \mathsf{FreqPub}$, the answer to Tqu is $\perp$ unless $(\langle \mathbf{g}, *\rangle, pk)$ occurs during $G^{\mathbf{O}}(S_{out})$. Now as before the idea is to show $\mathcal{B}$ can find S_0 and S_1 s.t. $S_{out} \in \{S_0, S_1\}$. To this end $\mathcal{B}$ starts simulating $D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots, C_n)$. By Lemma 16 all $\widetilde{\mathbf{g}}$ and $\widetilde{\mathbf{e}}$ queries can be handled. Let the sets W and W' be formed based on $\mathsf{Q_c}$ as in Lemma 4.8. For $\widetilde{\mathbf{d}}$

queries: $\mathcal{B}$ will be unable to simulate the answer to a query $qu = \widetilde{\mathbf{d}}(sk', c')$ only if $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$, $c' = c$ and $(pk, c) \notin \mathsf{W} \cup \mathsf{W'}$ (Case (c) of Lemma 16): in this case, knowing that the answer is the challenge bit b, $\mathcal{B}$ starts two branches of simulation, where it replies to qu with 0 on one branch and with 1 on the other. As before, we need to make sure $\mathcal{B}$ provides a consistent reply on either branch if the same query shows up in the future. The two strings decoded on the two branches at the end satisfy the above claim. □

To give the full proof of Theorem 18 we give a proposition which is an easy implication of Lemma 16.

Proposition 2. *Fix* $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w}, PK) \in Env(n)$ *and* $(\mathsf{Freq}, SK', \mathsf{Q_s}, \mathsf{Q_c}, \widetilde{\mathbf{g}}, \widetilde{\mathbf{e}}, \widetilde{\mathbf{d}}) \in Tvars^{\mathcal{O}}(PK)$. *Let* W *and* $\mathsf{W'}$ *be formed as in Lemma 16. For any* (sk, c) *the following holds:*

1. *If* $(\langle \mathfrak{g}, sk \rangle, *) \notin \mathsf{Q_s}$, *then* $\widetilde{\mathbf{d}}(sk, c)$ *can be efficiently computed given access to oracle* $\mathbf{O}$ *and having* $\mathsf{Q_c}$ *as input.*

2. *If* $(\langle \mathfrak{g}, sk \rangle, pk) \in \mathsf{Q_s}$, *for some* pk, *then* $\widetilde{\mathbf{d}}(sk, c)$ *can be efficiently computed given access to oracle* $\mathbf{O}$ *and by knowing* $\mathsf{Q_c}$ *and the value of* $\mathbf{u}(pk, c)$.

Theorem 18, (Restated). *Suppose* $\mathcal{A}$ *is a CCA2-valid adversary with access to oracles* $\mathcal{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{u}, \mathbf{w})$ *and* $\mathbf{T}$ *that makes at most* $2^{n/8}$ *queries. We have*

$$\Pr_{\mathcal{O}, \mathbf{T}, b, pk, c}[\mathcal{A}^{\mathcal{O}, \mathbf{T}}(1^n, pk, c) = b] \leq \frac{1}{2} + \frac{1}{2^{n/4}}, \tag{4.8.2}$$

where $\mathcal{O} \leftarrow \Psi$, $b \leftarrow \{0,1\}$, $sk \leftarrow \{0,1\}^n$, $pk = \mathbf{g}(sk)$ *and* $c \leftarrow \mathbf{e}(pk, b)$.

Proof. Fix $\mathcal{A}$ to be an adversary for which

$$\Pr_{\mathcal{O}, \mathbf{T}, b, pk, c}\left[\mathcal{A}^{(\mathcal{O}, \mathbf{T})}(1^n, pk, c) = b\right] = \frac{1}{2} + \alpha(n), \tag{4.8.3}$$

where all the variables are sampled as in Equation 4.8.2. We build a CCA2-valid adversary $\mathcal{B}$ with oracle access to $\mathcal{O} = (\mathbf{O}, \mathbf{u}, \mathbf{w})$ that on input $(1^n, pk, c)$ makes a list Que of at most $2^{n/4}$ queries and outputs b with probability at least $1/2 + \alpha(n)$. Note $\mathcal{B}$ should never call $\langle \mathbf{u}, (pk, c) \rangle$. The proof of Theorem 18 then follows from Lemma 10.

We now show how to build $\mathcal{B}$ based on $\mathcal{A}$. The adversary $\mathcal{B}^{\mathcal{O}}(1^n, pk, c)$ invokes $\mathcal{A}^{\mathcal{O}, \mathbf{T}}(1^n, pk, c)$, and reply to all $\mathcal{A}$'s $\mathcal{O}$ queries using its own $\mathcal{O}$ oracle. Note that since $\mathcal{A}$ is CCA2-valid it never makes the query $\langle \mathbf{u}, (pk, c) \rangle$.

To handle an $\mathcal{A}$'s query, $Tqu \stackrel{\text{def}}{=} \langle \mathbf{T}, (1^{n_1}, PK, C_1, \ldots, C_{n_1}) \rangle$, $\mathcal{B}$ acts as follows:

1. **Sampling** $\mathsf{Freq} \leftarrow FreqQue^{\mathbf{O},\mathbf{u}}(PK, n^{23})$: we show how $\mathcal{B}$ can do this. $\mathcal{B}^{\mathbf{O},\mathbf{u},\mathbf{w}}$ runs $E^{\mathbf{O}}(PK, 01)$ independently n^{23s} times and add the symbolic versions all the query/response pairs to Freq. Moreover, for any $(\langle \mathfrak{d}, (sk', c') \rangle, *) \in \mathsf{Freq}$, $\mathcal{B}$ calls $\langle \mathbf{g}, sk' \rangle$ to get pk': if $pk = pk'$ then $\mathcal{B}$ has found its challenge secret key so it halts and returns $\langle \mathbf{d}, (sk', c) \rangle$; otherwise, $\mathcal{B}$ calls $\langle \mathbf{u}, (pk', c') \rangle$: if the response is some $(b', r') \neq \perp$ then $\mathcal{B}$ adds $(\langle \mathfrak{e}, (pk, b', r') \rangle, c)$ to Freq. Finally, let $\mathsf{EmbedPut}$ contain any pk' such that $\langle \mathfrak{g}, * \rangle, pk') \in \mathsf{Freq}$ and assume without loss of generality that $pk \notin \mathsf{EmbedPub}$ (because otherwise $\mathcal{B}$ has found its challenge secret key and so it can halt and win the game).

2. $\mathcal{B}$ samples
$$(SK', \mathsf{Q_s}, \mathsf{Q_c}) \leftarrow ConsOrc(PK, \mathsf{Freq}), \tag{4.8.4}$$
This step is done offline.

3. $\mathcal{B}$ starts executing $D^{\widetilde{\mathbf{O}}}(SK', C_1 \ldots C_n)$, where
$$\begin{aligned}
&(\mathbf{e}_{imp}, \mathbf{d}_{imp}) = EncImpose(\mathbf{O}, \mathbf{u}, \mathsf{Q_c}, \mathsf{Q_{add}}, \mathsf{Q_{mult}}, \mathsf{Q_{rand}}),\\
&(\widetilde{\mathbf{g}}, \widetilde{\mathbf{d}}) = KeyImpose(\mathbf{g}, \mathbf{e}_{imp}, \mathbf{d}_{imp}, \mathsf{Q_s})\\
&\widetilde{\mathbf{e}} = \mathbf{e}_{imp} \text{ and } \widetilde{\mathbf{O}} = (\widetilde{\mathbf{g}}, \widetilde{\mathbf{e}}, \widetilde{\mathbf{d}}).
\end{aligned}$$

We explain how $\mathcal{B}$ can handle this decryption by considering the two possible sub-cases.

- $(\langle \mathfrak{g}, * \rangle, pk) \notin \mathsf{Q_s}$: we claim that $\mathcal{B}$ can successfully compute the decryption $D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C_{n_1})$. First, by Part (a) of Lemma 16, $\mathcal{B}$ can efficiently handle all $(\widetilde{\mathbf{g}}, \widetilde{e})$ queries asked during the execution. For a query $qu = \langle \widetilde{\mathbf{d}}, (sk', c') \rangle$ asked, either (i) $(\langle \mathfrak{g}, sk' \rangle, *) \notin \mathsf{Q_s}$ or (ii) for some $pk' \neq pk$, $(\langle \mathfrak{g}, sk' \rangle, pk') \in \mathsf{Q_s}$. $\mathcal{B}$ can handle Case (i) by Part 1 of Proposition 2, and Case (ii) by Part 2 of Proposition 2. Note that for Case (ii) $\mathcal{B}$ needs to make a query $\langle \mathbf{u}, (pk', c') \rangle$, which is a valid query for $\mathcal{B}$ since $pk \neq pk'$. Now if $S_{out} = D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C_{n_1})$, then $\mathcal{B}$ performs Step 5 of $\mathbf{T}$, which $\mathcal{B}$ can successfully do since the rest only involves making $\mathbf{g}$ and $\mathbf{w}$ queries. Thus, $\mathcal{B}$ can find the answer to Tqu.
- $(\langle \mathfrak{g}, * \rangle, pk) \in \mathsf{Q_s}$: In this case, recalling the definition of Qpub as given in $\mathbf{T}$'s description, we have $pk \in \mathsf{QPub}$, since $\mathbf{w}(pk) = \top$ and $|pk| \geq 5 \log n_1^{25s}$.

(The former is because pk is $\mathcal{B}$'s challenge public key which by definition is valid, and the latter is because $n_1 = poly(n)$.) Thus, by the condition given in Step 5 of $\mathbf{T}$'s description, if $S_{out} = D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C_{n_1})$ then at least one of the following conditions holds:

(a) The answer to Tqu is $\perp$; or

(b) $pk \in \mathsf{EmbedPub}$; or

(c) The query/response pair $(\langle \mathbf{g}, * \rangle, pk)$ will show up during $G^{\mathbf{O}}(S_{out})$.

We already know that $pk \notin \mathsf{EmbedPub}$ (see Step 1 of $\mathcal{B}$'s computation), thus if $S_{out} = D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C_{n_1})$ then one of Conditions (a) and (c) above holds.

We now claim the following.

Claim A. *If* $S_{out} = D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C_{n_1})$*, then* $\mathcal{B}$ can find two strings $S_{out,0}$ and $S_{out,1}$ such that $S_{out} \in \{S_{out,0}, S_{out,1}\}$.

If Claim A holds, $\mathcal{B}$ can execute both $G^{\mathbf{O}}(S_{out,0})$ and $G^{\mathbf{O}}(S_{out,1})$; if during either execution a query/response $(\langle \mathbf{g}, * \rangle, pk)$ is observed, $\mathcal{B}$ has learned $\mathbf{g}^{-1}(pk)$, winning the CCA2 game; otherwise, $\mathcal{B}$ in response to Tqu returns $\perp$, which is indeed the correct response. Thus, we show why Claim A holds.

Proof of Claim A. Assume

$$\mathsf{Q_c} = \{(\langle \mathfrak{e}, (pk_1, b_1, r_1) \rangle, c_1), \ldots, (\langle \mathfrak{e}, (pk_p, b_p, r_p) \rangle, c_p)\}.$$

$\mathcal{B}$ forms the two sets

$$\mathsf{W} = \{(pk_1, c_1), \ldots, (pk_p, c_p)\}$$
$$\mathsf{W}' = \{(pk_1, \mathbf{e}(pk_1, b_1, r_1)), \ldots, (pk'_p, \mathbf{e}(pk_p, b_p, r_p))\},$$

which $\mathcal{B}$ can efficiently do. Now $\mathcal{B}$ attempts to simulate the decryption $D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C_{n_1})$. By Part (a) of Lemma 16 $\mathcal{B}$ can efficiently reply to both $\widetilde{\mathbf{g}}$ and $\widetilde{\mathbf{e}}$ queries made along the way. For a query $qu = \widetilde{\mathbf{d}}(sk', c')$, one of the following possibilities holds:

- $(\langle \mathfrak{g}, sk' \rangle, *) \notin \mathsf{Q_s}$: in this case $\mathcal{B}$ can find the answer to qu by Part 1 of Proposition 2.
- $(\langle \mathfrak{g}, sk' \rangle, pk') \in \mathsf{Q_s}$ and $pk' \neq pk$: in this case $\mathcal{B}$ can find the answer to qu by Part 2 of Proposition 2.

- $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$ and $c' \in \mathsf{W} \cup \mathsf{W}'$: in this case, by Parts (d) and (e) of Lemma 16, $\mathcal{B}$ can determine the answer to qu.
- $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$, $c' \neq c$ and $c' \notin \mathsf{W} \cup \mathsf{W}'$: in this case by Part (c) of Lemma 16 the answer to qu can be determined by calling $\langle \mathbf{u}, (pk, c') \rangle$, which is a valid query for $\mathcal{B}$, as $c' \neq c$.
- $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$, $c' = c$ and $c \notin \mathsf{W} \cup \mathsf{W}'$: in this case by Part (c) of Lemma 16 the answer to qu is the unknown challenge bit b. Thus, for the first time this case occurs, $\mathcal{B}$ starts two parallel branches BR_0 and BR_1 of computation, where $\mathcal{B}$ replies to qu with b_1 on branch BR_{b_1}. On both branches $\mathcal{B}$ replies to all $\widetilde{\mathbf{g}}$, $\widetilde{\mathbf{e}}$ and $\widetilde{\mathbf{d}}$ exactly as explained above. Moreover, if on some branch $BR_{b'}$, still during the execution of $D^{\widetilde{\mathbf{O}}}(SK', C_1, \ldots, C'_{n_1})$, the current query is asked again (i.e., the query $qu' = \langle \widetilde{\mathbf{d}}, (sk', c') \rangle$, where $(\langle \mathfrak{g}, sk' \rangle, pk) \in \mathsf{Q_s}$ and $c' \notin \mathsf{W} \cup \mathsf{W}'$) $\mathcal{B}$ replies to qu' with b', making it consistent with the previous reply.

Claim A now follows by the above way of simulation.

At the end of $\mathcal{A}$'s simulation $\mathcal{B}$ outputs whatever $\mathcal{A}$ does. Note that $\mathcal{B}$ makes at most $2^{n/8} \times poly(n) \leq 2^{n/4}$ queries, since $\mathcal{B}$ replies to each $\mathcal{A}$'s query by making $poly(n)$ queries. Moreover, note that whenever $\mathcal{B}$ halts without completing the stimulation, it has found the challenge bit b. The proof of Theorem 18 is now complete. □

4.9 Establishing the Separation Result

In this section we show how to combine our results from Sections 4.7 and 4.8 to establish our separation result. We first begin with the following lemma.

Lemma 17. *For any construction (G, E, D), there exists a PPT oracle adversary $\mathcal{A}$, for which*

$$\Pr_{\mathbf{O},\mathbf{T}}\left[\mathcal{A}^{\mathbf{O},\mathbf{T}} \textit{ breaks the weak-seed-circular security of } (G^{\mathbf{O}}, E^{\mathbf{O}}, D^{\mathbf{O}})\right] = 1 \qquad (4.9.1)$$

Proof. First, note by Theorem 16 there exists a PPT oracle adversary $\mathcal{A}$ s.t.

$$\Pr_{\mathbf{O},\mathbf{T},S,(C_1,\ldots,C_n)}\left[\mathcal{A}^{\mathbf{O},\mathbf{T}}(PK, C_1, \ldots, C_n) = S\right] \geq 1 - \frac{1}{n^5},$$

where $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d}) \leftarrow \Psi$, $S \leftarrow \{0,1\}^n$, $(PK, SK) = G^{\mathbf{O}}(S)$ and $(C_1, \ldots, C_n) \leftarrow E^{\mathbf{O}}(PK, S)$.

Using a simple averaging argument we obtain

$$\Pr_{\mathbf{O},\mathbf{T}}\left[\Pr_{S,(C_1,\ldots,C_n)}\left[\mathcal{A}^{\mathbf{O},\mathbf{T}}(PK, C_1, \ldots, C_n) = S\right] \geq 1 - \frac{1}{n^3}\right] \geq 1 - \frac{1}{n^2}.$$

This means that for at most a $\frac{1}{n^2}$ fraction of $(\mathbf{O}, \mathbf{T})$ the adversary $\mathcal{A}^{\mathbf{O},\mathbf{T}}$, on security parameter 1^n, outputs S with probability less than $1 - \frac{1}{n^3}$. Since $\sum_{n=0}^{\infty} \frac{1}{n^2}$ converges, by the Borel-Cantelli lemma we obtain that for measure-one of oracles $(\mathbf{O}, \mathbf{T})$ the adversary $\mathcal{A}^{\mathbf{O},\mathbf{T}}$ breaks the weak-seed-circular security of $(G^{\mathbf{O}}, E^{\mathbf{O}}, D^{\mathbf{O}})$: for all sufficiently large n the adversary $\mathcal{A}^{\mathbf{O},\mathbf{T}}$ recovers S from $(PK, C_1, \ldots, C_n)$ with probability at least $1 - \frac{1}{n^3}$. □

Theorem 19. *There exists no fully-black-box construction of weak-seed-circular-secure bit-encryption schemes from CCA2-secure encryption] schemes.*

Proof. Suppose to the contrary that such a reduction exists and let (G, E, D) and *Red* be, respectively, the associated construction and the security-proof algorithms. By Lemma 17 there exists a PPT-oracle adversary $\mathcal{A}$ such that

$$\Pr_{\mathbf{O}=(\mathbf{g},\mathbf{e},\mathbf{d}),\mathbf{T}}\left[\mathcal{A}^{\mathbf{O},\mathbf{T}} \text{ breaks the weak-seed-circular security of } (G^{\mathbf{O}}, E^{\mathbf{O}}, D^{\mathbf{O}})\right] = 1.$$

From above we obtain there exists a PPT-oracle adversary $\mathcal{A}$ such that for a measure one of oracles $\mathcal{OR}$ we have that for any $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{T}) \in \mathcal{OR}$, it holds that $\mathcal{A}^{\mathbf{O},\mathbf{T}}$ breaks the weak-seed-circular security of $(G^{\mathbf{O}}, E^{\mathbf{O}}, D^{\mathbf{O}})$. Now, letting $\alpha(n) = n^{-\log n}$ (it can be any value so long as $\frac{1}{\alpha(n)}$ is sub-exponential) we obtain that for any $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{T}) \in \mathcal{OR}$, it holds that

$$\Pr\left[Red^{\left(\mathcal{A}^{\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{T}},\mathbf{u}\right)}(1^n, pk, c) = b\right] \geq \frac{1}{2} + \alpha(n), \tag{4.9.2}$$

where $b \leftarrow \{0,1\}$, $sk \leftarrow \{0,1\}^n$, $pk = \mathbf{g}(sk)$ and $c \leftarrow \mathbf{e}(pk, b)$. The oracle $\mathbf{u}$ is provided to *Red* because by definition *Red* is an adversary against CCA2 security. (Also, note that *Red* never calls $\langle \mathbf{u}, (pk, c) \rangle$.) Now since both *Red* and $\mathcal{A}$ are PPT, there exists a PPT CCA2-valid adversary $\mathcal{B}$ such that for any $(\mathbf{g}, \mathbf{e}, \mathbf{d}, \mathbf{T}) \in \mathcal{OR}$

$$\Pr_{b,pk,c}\left[\mathcal{B}^{\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{u},\mathbf{T}}(1^n, pk, c) = b\right] \geq \frac{1}{2} + \alpha(n). \tag{4.9.3}$$

Now since $\mathcal{OR}$ is a measure-one subset of the set of all oracles, we have

$$\Pr_{\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{u},\mathbf{w},\mathbf{T},b,pk,c}\left[\mathcal{B}^{\mathbf{O},\mathbf{T}}(1^n,pk,c)=b\right]\geq\frac{1}{2}+\alpha(n),$$

where $(\mathbf{g},\mathbf{e},\mathbf{d},\mathbf{u},\mathbf{w})\leftarrow\Psi$ and others are sampled as above. However, this contradicts Theorem 18, and the proof is complete. □

4.10 Constructions based on Circular Security

We show how our results on seed-circular security extend to rule out a class of constructions for circular security that we call *key-isolating constructions*. To define this class we first define it in a related model we call the *canonical model*, and then we define it in the standard model. We start with some definitions.

Canonical-Form (CF) PKE. We call $\mathbf{O}=(\mathbf{gs},\mathbf{gp},\mathbf{e},\mathbf{d})$ a CF PKE if the domain of $\mathbf{gp}$ (excluding 1^n) is the range of $\mathbf{gs}$ and $(\mathbf{g},\mathbf{e},\mathbf{d})$, where $\mathbf{g}(s)=(\mathbf{gp}(\mathbf{gs}(s)),\mathbf{gs}(s))$, is a PKE. That is, the key-generation algorithm of a CF scheme first deterministically maps a seed to a secret key, and then *deterministically* maps the secret key to a public key.

CF-based black-box model. A black-box construction in the CF model is a tuple of oracle algorithms (GS,GP,E,D) s.t. for any CF PKE $\mathbf{O}=(\mathbf{gs},\mathbf{gp},\mathbf{e},\mathbf{d})$, $(GS^{\mathbf{O}},GP^{\mathbf{O}},E^{\mathbf{O}},D^{\mathbf{O}})$ is a CF PKE. Proving a syntactically-unrestricted impossibility result in the CF model implies one in the standard model, since any CPA-secure CF PKE can be turned into a CPA-secure standard PKE and that any circular-secure standard PKE can be put into a circular-secure CF PKE.

CF Key-isolating constructions. We call (GS,GP,E,D) *key-isolating* if GS never calls $\mathbf{gp}$ of the base scheme, i.e., GS only has access to $(\mathbf{gs},\mathbf{e},\mathbf{d})$.

Ruling-out key-isolating constructions. Our earlier results extend to rule out CF key-isolating constructions for circular security. To do this, we first need to change the distribution of Ψ, by replacing $\mathbf{g}$ with $(\mathbf{gs},\mathbf{gp})$, for $\mathbf{gs}_n\colon\{0,1\}^n\to\{0,1\}^{3n}$ and $\mathbf{gp}_n\colon\{0,1\}^{3n}\to\{0,1\}^{5n}$. As for $\mathbf{T}$, which now takes as input a public key and an encryption of a PK's secret key, all we need to change is that in Step 5 of $\mathbf{T}$'s description the set $\mathsf{EmbedPub}$ should be formed by executing $GP^{\mathbf{O}}$ on the intermediate, decrypted string (which is now a secret key). All our proofs about $\mathbf{T}$ not breaking the semantic security of the base scheme go through with only making

obvious modifications. The proofs about $\mathbf{T}$ being helpful in breaking the circular security of the constructed scheme follow by noting that all access to $\mathbf{gp}$ during key generation is only made by GP. This fact only becomes essential in the proof of Lemma 12, and is the reason behind the above way of defining $\mathsf{EmbedPub}$. Other lemmas follow by making only obvious changes.

Interpretation with respect to standard constructions. Our above result also rules out *standard key-isolating* constructions. To define this notion for a standard construction (G, E, D) we first need to slightly change the standard model so that (G, E, D) takes as oracles a CF PKE. Again, as explained above this is w.l.o.g. Now we call $\mathcal{E} = (G, E, D)$ a *standard key-isolating construction* if $\mathcal{E}$ admits a key-isolating CF *counterpart* in the following sense: there exists algorithms GS and GP s.t. (GS, GP, E, D) is key-isolating and (GS, GP) induces the same distribution as G, i.e., for any $\mathbf{O} = (\mathbf{gs}, \mathbf{gp}, \mathbf{e}, \mathbf{d})$ it holds that (PK, SK) is identically distributed to (PK', SK'), where $(PK, SK) \leftarrow G^{\mathbf{O}}(1^n)$, $SK' \leftarrow GS^{\mathbf{gs},\mathbf{e},\mathbf{d}}(1^n)$ and $PK' = GP^{\mathbf{O}}(SK')$. Now the impossibility of CF key-isolating constructions extends to standard ones, by how the notion counterpart is defined.

Examples. Any standard construction $\mathcal{E} = (G, E, D)$ under which seeds and secret keys are the same is key-isolating: defining $GS(S) = S$ and $GP^{\mathbf{gs},\mathbf{gp},\mathbf{e},\mathbf{d}}(S) = G_2^{\mathbf{gs},\mathbf{gp},\mathbf{e},\mathbf{d}}(S)$, where G_2 is the algorithm corresponding to the public-key output of G, the construction (GS, GP, E, D) is the CF-counterpart of (G, E, D) and is key-isolating since GS makes no oracle calls at all. The class of key-isolating constructions is larger than this; we only wanted to give a concrete example.

4.11 Toward an Impossibility Result for Circular Security

In this section we briefly explain why we were not able to fully extend our results to the circular-security case. For simplicity, we highlight the difficulties encountered with respect to the simple type of constructions discussed in Section 4.5. In what follows all mentions of the oracle $\mathbf{T}$ for the seed-circular-security case refers to the oracle $\mathbf{T}$ defined in Section 4.5.

As discussed previously, the main challenge in designing an appropriate Oracle $\mathbf{T}$ is to make sure that responses to queries to $\mathbf{T}$ do not leak information about the challenge secrets of a CPA adversary $\mathcal{A}$ against $\mathbf{O} = (\mathbf{g}, \mathbf{e}, \mathbf{d})$. We proved this for

the seed-circular-security case by providing a CCA2 adversary $\mathcal{B}$ in such a way that $\mathcal{B}^{\mathbf{O},\mathbf{u},\mathbf{w}}(pk, c)$ is able to simulate $\mathcal{A}^{\mathbf{O},\mathbf{T}}(pk, c)$.

Roughly speaking, the only part of the execution of $\mathbf{T}(PK, C_1, \ldots, C_n)$ that is not simulatable by a CCA2-adversary $\mathcal{B}^{\mathbf{O},\mathbf{u},\mathbf{w}}(pk, c)$ is when during the computation of $D^{\widetilde{\mathbf{d}}}(SK', C_1 \ldots C_n)$ a query $\widetilde{\mathbf{d}}(sk', c)$ shows up and $(\langle \mathbf{g}, sk' \rangle, pk) \in \mathsf{Q_s}$. We fixed this non-simulatability problem by adding an extra check at the end of $\mathbf{T}$'s computation that ensures the following: either the value of $\mathbf{g}^{-1}(pk)$ is *embedded* in S_{out} (i.e., the query/response pair $(\langle \mathbf{g}, * \rangle, pk)$ shows up during $G^{\mathbf{g}}(S_{out})$) or the answer to the underlying $\mathbf{T}$ query is $\perp$.

To define a circular-security weakening oracle $\mathbf{T}$ we may be tempted to proceed as before: $\mathbf{T}$ accepts inputs of the form $(PK, C_1, \ldots, C_n)$, where now $C_1, \ldots, C_n$ are (supposedly) bit-wise encryption of a PK's secret key under PK itself. (For simplicity, assume that the length of the secret key is n.) Then, everything remain unchanged, as in $\mathbf{T}$ in Section 4.5, until $\mathbf{T}$ obtains $S_{out} = D^{\widetilde{\mathbf{d}}}(SK', C_1 \ldots C_n)$, which now is supposedly a PK's matching secret key. Now in order to make sure that the oracle $\mathbf{T}$ is simulatable (i.e., it does not leak non-simulatable information to a CPA adversary against $\mathbf{O}$) it seems that, as before, we need to make sure that $\mathbf{g}^{-1}(\mathsf{Q_{pub}})$ is "embedded" in S_{out}, before releasing S_{out}. (Recall the definition of $\mathsf{Q_{pub}}$ from $\mathbf{T}$'s definition in Section 4.5.) But this "embedding condition" seems hard to check. This check was easy for the seed-circular-security case since we can simply run $G^{\mathbf{g}}(S_{out})$ and monitor all sk' for which we observe a query/response $(\langle \mathbf{g}, sk' \rangle, *)$. For the circular-security case one idea is to run $D^{\mathbf{d}}(S_{out}, \cdot)$ on many random encryptions produced as $C \leftarrow E^{\mathbf{e}}(PK, b; R)$ for randomly chosen b and R, and record in a set $\mathsf{EmbedSec}$ all sk' for which we encounter a query $\langle \mathbf{d}, (sk', *) \rangle$. We then return S_{out} if $\mathsf{Q_{pub}} \subseteq \mathbf{g}(\mathsf{EmbedSec})$; otherwise, we return $\perp$. While this check makes the oracle $\mathbf{T}$ simulatable, it makes $\mathbf{T}$ unfortunately too weak in that we cannot anymore guarantee in general that $\mathbf{T}(PK, C_1 \ldots C_n)$ will return SK with non-negligible probability, for $(PK, SK) \leftarrow G^{\mathbf{g}}(1^n)$ and $(C_1, \ldots, C_n) \leftarrow E^{\mathbf{e}}(PK, SK)$, i.e., $\mathbf{T}$ is not useful in general for breaking circular security. (Contrived constructions (G, E, D) for which this is the case can be given.)

4.12 Open Problems

The main open problem is to extend our impossibility result to all constructions for circular security. We explained in Section 4.11 why we were not able to do this.

Another interesting problem is to see to what extent our techniques extend to obtain impossibility results based on other classical public-key primitives, e.g., trapdoor permutations. Finally, we conjecture that *k-circular security* (i.e., security under an encrypted cycle involving k pairs of public/secret keys) does not imply $(k+1)$-circular security in a black-box way, for any $k \geq 0$.

Bibliography

[1] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *Theory of Cryptography*, pages 474–495. Springer, 2009.

[2] Joël Alwen, Yevgeniy Dodis, Moni Naor, Gil Segev, Shabsi Walfish, and Daniel Wichs. Public-key encryption in the bounded-retrieval model. In *Advances in Cryptology–EUROCRYPT 2010*, pages 113–134. Springer, 2010.

[3] Benny Applebaum. Key-dependent message security: Generic amplification and completeness. *Journal of Cryptology*, 27(3):429–451, 2014.

[4] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology-CRYPTO 2009*, pages 595–618. Springer, 2009.

[5] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In *FOCS, 2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 191–209. IEEE, 2015.

[6] Gilad Asharov and Gil Segev. On constructing one-way permutations from indistinguishability obfuscation. In *Theory of Cryptography*, pages 512–541. Springer, 2016.

[7] Paul Baecher, Christina Brzuska, and Marc Fischlin. Notions of black-box reductions, revisited. In *Advances in Cryptology-ASIACRYPT 2013*, pages 296–315. Springer Berlin Heidelberg, 2013.

[8] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im) possibility of obfuscating programs. *Journal of the ACM (JACM)*, 59(2):6, 2012.

[9] Boaz Barak, Iftach Haitner, Dennis Hofheinz, and Yuval Ishai. Bounded key-dependent message security. In *Advances in Cryptology–EUROCRYPT 2010*, pages 423–444. Springer, 2010.

[10] Mihir Bellare, Alexandra Boldyreva, and Adam ONeill. Deterministic and efficiently searchable encryption. In *Advances in Cryptology-CRYPTO 2007*, pages 535–552. Springer, 2007.

[11] Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. Randomness re-use in multi-recipient encryption schemeas. In *Public Key CryptographyPKC 2003*, pages 85–99. Springer, 2003.

[12] Mihir Bellare, Marc Fischlin, Adam ONeill, and Thomas Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *Advances in Cryptology–CRYPTO 2008*, pages 360–378. Springer, 2008.

[13] Eleanor Birrell, Kai-Min Chung, Rafael Pass, and Sidharth Telang. Randomness-dependent message security. In *Theory of Cryptography*, pages 700–720. Springer, 2013.

[14] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *SAC 2002, Selected Areas in Cryptography*, volume 2595 of *LNCS*, pages 62–75. Springer, 2002.

[15] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM Journal on Computing*, 20(6):1084–1118, 1991.

[16] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 103–112. ACM, 1988.

[17] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM journal on Computing*, 13(4):850–864, 1984.

[18] Alexandra Boldyreva, Serge Fehr, and Adam ONeill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *Advances in Cryptology–CRYPTO 2008*, pages 335–359. Springer, 2008.

[19] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2006.

[20] Dan Boneh, Shai Halevi, Mike Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *Advances in Cryptology–CRYPTO 2008*, pages 108–125. Springer, 2008.

[21] Dan Boneh, Periklis A Papakonstantinou, Charles Rackoff, Yevgeniy Vahlis, and Brent Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 283–292. IEEE, 2008.

[22] Joan F Boyar, Stuart A Kurtz, and Mark W Krentel. A discrete logarithm implementation of perfect zero-knowledge blobs. *Journal of Cryptology*, 2(2):63–76, 1990.

[23] Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability. In *Advances in Cryptology–CRYPTO 2010*, pages 1–20. Springer, 2010.

[24] Zvika Brakerski, Shafi Goldwasser, and Yael Tauman Kalai. Black-box circular-secure encryption beyond affine functions. In *Theory of Cryptography*, pages 201–218. Springer, 2011.

[25] Zvika Brakerski, Jonathan Katz, Gil Segev, and Arkady Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In *Theory of Cryptography*, pages 559–578. Springer, 2011.

[26] Zvika Brakerski and Gil Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. *Journal of cryptology*, 27(2):210–247, 2014.

[27] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[28] Jan Camenisch, Nishanth Chandran, and Victor Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *LNCS*, pages 351–368. Springer, 2009.

[29] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology (EUROCRYPT 2001)*, pages 93–118. Springer, 2001.

[30] Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors. *Public-Key Cryptography - PKC 2016*.

[31] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *Theory of Cryptography*, pages 427–444. Springer, 2008.

[32] Seung Geol Choi and Hoeteck Wee. Lossy trapdoor functions from homomorphic reproducible encryption. *Information Processing Letters*, 112(20):794–798, 2012.

[33] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Advances in Cryptology (Eurocrypt 2002)*, pages 45–64. Springer, 2002.

[34] Ivan B Damgård, Torben P Pedersen, and Birgit Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. *Journal of Cryptology*, 10(3):163–194, 1997.

[35] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *Theory of Cryptography*, pages 361–381. Springer, 2010.

[36] Yevgeniy Dodis, Yael Tauman Kalai, and Shachar Lovett. On cryptography with auxiliary input. In *STOC 2009*, pages 621–630, 2009.

[37] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM journal on computing*, 38(1):97–139, 2008.

[38] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC 1991*, pages 542–552, 1991.

[39] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS 2008*, pages 293–302, 2008.

[40] U Feige, D Lapidot, and A Shamir. Multiple non-interactive zero knowledge proofs based on a single random string. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, pages vol–1. IEEE Computer Society, 1990.

[41] David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. *J. Cryptology*, 26(1):39–74, 2013.

[42] Benjamin Fuller, Adam ONeill, and Leonid Reyzin. A unified approach to deterministic encryption: New constructions and a connection to computational entropy. *Journal of Cryptology*, 28(3):671–717, 2015.

[43] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing-STOC'09*, pages 169–169. ACM Press, 2009.

[44] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206. ACM, 2008.

[45] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 99–108. ACM, 2011.

[46] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 325–335. IEEE, 2000.

[47] Yael Gertner, Tal Malkin, and Steven Myers. Towards a separation of semantic and CCA security for public key encryption. In *Theory of Cryptography*, pages 434–455. Springer, 2007.

[48] Yael Gertner, Tal Malkin, and Omer Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *FOCS*, page 126. IEEE, 2001.

[49] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In *Advances in Cryptology*, pages 276–288. Springer, 1984.

[50] Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32. ACM, 1989.

[51] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229. ACM, 1987.

[52] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM (JACM)*, 38(3):690–728, 1991.

[53] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[54] Shafi Goldwasser, Silvio Micali, and Po Tong. Why and how to establish a private code on a public network. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, pages 134–144. IEEE Computer Society, 1982.

[55] Iftach Haitner and Thomas Holenstein. On the (im) possibility of key dependent encryption. In *Theory of Cryptography*, pages 202–219. Springer, 2009.

[56] Iftach Haitner and Omer Reingold. Statistically-hiding commitment from any one-way function. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 1–10. ACM, 2007.

[57] Mohammad Hajiabadi and Bruce M Kapron. Reproducible circularly-secure bit encryption: Applications and realizations. In *Advances in Cryptology–CRYPTO 2015*, pages 224–243. Springer, 2015.

[58] Mohammad Hajiabadi and Bruce M. Kapron. Toward separating circular-secure bit encryption from semantically-secure encryption. 2016.

[59] Mohammad Hajiabadi, Bruce M. Kapron, and Venkatesh Srinivasan. On generic constructions of circularly-secure, leakage-resilient public-key encryption schemes. In Cheng et al. [30], pages 129–158.

[60] Shai Halevi and Huijia Lin. After-the-fact leakage in public-key encryption. In *Theory of Cryptography*, pages 107–124. Springer, 2011.

[61] Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[62] Brett Hemenway and Rafail Ostrovsky. Building injective trapdoor functions from oblivious transfer. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:127, 2010.

[63] Brett Hemenway and Rafail Ostrovsky. Building lossy trapdoor functions from lossy encryption. In *Advances in Cryptology-ASIACRYPT 2013*, pages 241–260. Springer, 2013.

[64] Dennis Hofheinz. Circular chosen-ciphertext security with compact ciphertexts. In *Advances in Cryptology–EUROCRYPT 2013*, pages 520–536. Springer, 2013.

[65] Dennis Hofheinz and Dominique Unruh. Towards key-dependent message security in the standard model. In *Advances in Cryptology–EUROCRYPT 2008*, pages 108–126. Springer, 2008.

[66] Chun-Yuan Hsiao and Leonid Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Advances in Cryptology–CRYPTO 2004*, pages 92–105. Springer, 2004.

[67] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 44–61. ACM, 1989.

[68] Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 2006.

[69] Mohammad Mahmoody and Rafael Pass. The curious case of non-interactive commitments–on the power of black-box vs. non-black-box use of primitives. In *Advances in Cryptology–CRYPTO 2012*, pages 701–718. Springer, 2012.

[70] Tal Malkin, Isamu Teranishi, and Moti Yung. Efficient circuit-size independent public key encryption with kdm security. In *Advances in Cryptology – EUROCRYPT 2011*, pages 507–526, 2011.

[71] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In *Theory of Cryptography, TCC 2004*, volume 2951 of *LNCS*, pages 278–296. Springer, 2004.

[72] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1995.

[73] Steven Myers and Abhi Shelat. Bit encryption is complete. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 607–616. IEEE, 2009.

[74] Moni Naor, Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Perfect zero-knowledge arguments for np using any one-way permutation. *Journal of Cryptology*, 11(2):87–108, 1998.

[75] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. *SIAM Journal on Computing*, 41(4):772–814, 2012.

[76] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 33–43. ACM, 1989.

[77] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 427–437. ACM, 1990.

[78] Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

[79] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology (EUROCRYPT99)*, pages 223–238. Springer, 1999.

[80] Rafael Pass. Limits of provable security from standard assumptions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 109–118. ACM, 2011.

[81] Rafael Pass. Unprovable security of perfect NIZK and non-interactive non-malleable commitments. In *Theory of Cryptography*, pages 334–354. Springer, 2013.

[82] Rafael Pass, abhi shelat, and Vinod Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. *Advances in Cryptology-CRYPTO 2006*, pages 271–289, 2006.

[83] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM Journal on Computing*, 40(6):1803–1844, 2011.

[84] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO 1991*, volume 576 of *LNCS*, pages 433–444. Springer, 1991.

[85] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[86] Omer Reingold, Luca Trevisan, and Salil Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of Cryptography*, pages 1–20. Springer, 2004.

[87] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 387–394. ACM, 1990.

[88] Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. *SIAM Journal on Computing*, 39(7):3058–3088, 2010.

[89] Ron D Rothblum. On the circular security of bit-encryption. In *Theory of Cryptography*, pages 579–598. Springer, 2013.

[90] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 475–484. ACM, 2014.

[91] Daniel R Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology (EUROCRYPT'98)*, pages 334–345. Springer, 1998.

[92] Yevgeniy Vahlis. Two is a crowd? a black-box separation of one-wayness and security under correlated inputs. In *Theory of Cryptography*, pages 165–182. Springer, 2010.

[93] Hoeteck Wee. Dual projective hashing and its applicationslossy trapdoor functions and more. In *Advances in Cryptology–EUROCRYPT 2012*, pages 246–262. Springer, 2012.

[94] Hoeteck Wee. KDM-security via homomorphic smooth projective hashing. In Cheng et al. [30], pages 159–179.

[95] Andrew C Yao. Theory and application of trapdoor functions. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, pages 80–91. IEEE Computer Society, 1982.