Universal Amplification of KDM Security: From 1-Key Circular to Multi-Key KDM

Brent Waters [*]
UT Austin and NTT Research

Daniel Wichs[†]
Northeastern and NTT Research

July 6, 2023

Abstract

An encryption scheme is *Key Dependent Message* (KDM) secure if it is safe to encrypt messages that can arbitrarily depend on the secret keys themselves. In this work, we show how to upgrade essentially the weakest form of KDM security into the strongest one. In particular, we assume the existence of a *symmetric-key bit-encryption* that is *circular-secure* in the 1-*key* setting, meaning that it maintains security even if one can encrypt individual bits of a single secret key under itself. We also rely on a standard CPA-secure public-key encryption. We construct a *public-key* encryption scheme that is *KDM secure for general functions* (of a-priori bounded circuit size) in the *multi-key* setting, meaning that it maintains security even if one can encrypt arbitrary functions of arbitrarily many secret keys under each of the public keys. As a special case, the latter guarantees security in the presence of arbitrary length key cycles. Prior work already showed how to amplify n-key circular to n-key KDM security for general functions. Therefore, the main novelty of our work is to upgrade from 1-key to n-key security for arbitrary n.

As an independently interesting feature of our result, our construction does not need to know the actual *specification* of the underlying 1-key circular secure scheme, and we only rely on the *existence* of some such scheme in the proof of security. In particular, we present a *universal* construction of a multi-key KDM-secure encryption that is secure as long as some 1-key circular-secure scheme exists. While this feature is similar in spirit to Levin's universal construction of one-way functions, the way we achieve it is quite different technically, and does not come with the same "galactic inefficiency".

1 Introduction

An encryption system is Key Dependent Message (KDM) secure if it maintains security even if the messages can arbitrarily depend on the secret key. KDM security has several interesting connections to both the application and theory of cryptography. For instance, the Bitlocker encryption utility will sometimes encrypt its own secret key resulting in a KDM ciphertext. The anonymous credential system of Camenisch and Lysyanskaya [CL01] requires users encrypt their own secret keys in an effort to deter users from delegating their keys. More recently the work of Koppula and Waters [KW19] showed how to build chosen ciphertext security [RS92] PKE via a key dependent

[*]Supported by NSF CNS-1908611 and Simons Investigator award.
[†]Research supported by NSF CNS-1750795, CNS-2055510, and the JP Morgan faculty research award.

type primitive they called hinting PRG. Their methods were subsequently adapted [KMT19] to apply a similar transformation using KDM secure encryption. Following this [LQR$^+$19, KM19] showed how to leverage these techniques to build resuable designated verifier zero knowledge proofs. In addition, earlier connections to CCA were made [HK15], but using an additional reproducibility property.

On the flipside one can gain many insights from the technically challenging problem of creating cryptosystems that meet the traditional notion of CPA security [GM82], but are *not* KDM secure. Multiple works [ABBC10, CGH12, Rot13, BHW15, KRW15, KW16, AP16, GKW17b, GKW17c] have explored such questions employing a variety of number theoretic assumptions. Notably, the KDM insecure bit encryption system of [GKW17c] forms the machinery upon which lockable or compute and compare obfuscation is built [GKW17a, WZ17].

Initially, the only provably KDM secure constructions of encryption systems relied on the random oracle heuristic [BRS03]. In a breakthrough result Boneh, Halevi, Hamburg and Ostrovsky [BHHO08] showed how to achieve public key encryption system that were n-key KDM secure for affine functions under the Decisional Diffie-Hellman (DDH) assumption. At a very high level their solution consisted to two core ideas. First, they manipulated algebraic properties of DDH hard groups to allow for the creation of a ciphertext with a message that depended on a chosen affine function of a secret key, but without knowing the secret key. Second, they created a proof technique that allowed for a reduction to (undetectably) derive several public keys from one "master" public key. This second idea is what allowed them to get KDM security for n keys. It also relied on the underlying algebraic properties of the groups. Subsequently, [ACPS09] showed how these concepts applied in domains beyond DDH hard groups. They achieved n-KDM security for affine functions from the Learning with Errors (LWE) assumption for public key encryption and from the Learning Parity with Noise (LPN) assumption for secret key encryption. The work of [BLSV18] showed how to construct 1-KDM security for affine functions generically from a building block called *batch encryption* (a variant of chameleon encryption [DG17] and laconic OT [CDG$^+$17]), which can be instantiated from the CDH assumption, the LWE assumption, or LPN with extremely small noise.

At first glance having KDM security for just affine functions seems limited as one would ideally like to achieve security for arbitrary functions of the secret key. To this end a second set of works [BHHI10, BGK11, App11, KM20] looked to amplify security from weak to more expressive functions. Barak et al. [BHHI10] first showed how to get n-key KDM security for functions of bounded circuit-size from the DDH or LWE assumptions. Applebaum [App11] then showed a generic technique for amplifying *any* n-key KDM security for *projection functions* to n-key KDM security to arbitrary functions of bounded circuit-size using garbled circuits.[1] Projection functions are intuitively functions where every output bit depends on at most a single bit from the secret keys. That is an output bit can be a constant 0 or 1 bit, one that copies a key bit, or one that flips it. Critically, projections are captured by the class of affine functions from the work of [BHHO08] and [ACPS09]. Kitagawa and Matsuda [KM20] later employed garbling to amplify from an even weaker class of functionality, which allowed for constant functions or copying a secret key bit, but not flipping it, which they call *circular security*. (We note in the KDM literature the term "circular security" can also refer a different notion of just encrypting key cycles.)

The most recent works give an eloquent and generic way to move from a scheme supporting

[1] Applebaum [App11] describes his transformation using the terminology of randomized encodings; however, we refer to it in terms of garbled circuits to simplify comparisons.

a very weak form of KDM queries to an expressive over a given number of keys. However, there do not exist techniques that allow for amplification of the *number of keys* a KDM supports. These techniques are essentially "stuck with" whatever number of keys are supported in the base scheme leaving fundamental questions.[2] For example, if we start with a 1-KDM secure scheme can we achieve KDM security for n-key cycles for $n > 1$? Or arbitrary functions on n keys?

Our Results. In this work we answer these questions in the affirmative. Starting with any standard CPA-secure public key encryption (PKE), we create a PKE that is n-key KDM secure for arbitrary n and for any functions of a-priori bounded circuit size, as long as there exists some symmetric-key encryption scheme that is 1-key circular secure. As a concrete corollary of our work, by relying on the existence of 1-key circular secure encryption from the CDH assumption [BLSV18], we get the first n-key KDM secure scheme under CDH.

Like prior works [BHHI10,App11,KM20] our construction leverages garbled circuits, but with a different twist. Remarkably, our construction does need to know the specification of any 1-key circular secure scheme. Instead the 1-key circular scheme only manifests in our proof of security. Thus we are able to achieve a form of *universal security*, where our concrete scheme is secure so long as there exists some 1-key circular secure scheme, without needing to know which one.[3] Even though our construction is universal, it does not employ any "algorithm enumeration" style techniques associated with most other universal constructions starting with [Lev85]. It also avoids the "galactic inefficiency" tied to such approaches, where huge constant factors in the resulting schemes imply that they likely cannot be executed in the lifetime of the universe if we want reasonable concrete security. Interestingly, our proof would still work even if the underlying 1-key circular scheme only had a non-uniform instantiation, meaning that the description of the algorithms themselves could depend on λ but their specification could take superpolynomial time to generate. This is in contrast to other prior universal constructions based on "algorithm enumeration", which crucially rely on the underlying secure scheme having a uniform (constant-sized) algorithmic description.

A core contribution is that we devise a proof method to link several disparate secret keys to a single master key. At the beginning of the proof all n secret keys are independently generated, but at the end they can all be viewed as being derived from an introduced master key. This matches in spirit the key linking concept pioneered by [BHHO08], but with the difference that we generically utilize 1-key circular security to achieve the linking as opposed to algebraic properties of a particular number theoretic domain.

1.1 Overview

We now give an overview of our construction and proof of security. A feature of our construction is its simplicity and we begin by describing the algorithms. We will utilize a CPA-secure public-key encryption scheme $\mathsf{CPA} = (\mathsf{CPA.KeyGen}, \mathsf{CPA.Enc}, \mathsf{CPA.Dec})$ that is not necessarily KDM secure and a garbled circuit scheme $(\mathsf{Garble}, \mathsf{Eval})$. In the security proof, we will further assume the existence of some symmetric-key encryption scheme that is 1-key KDM secure for

[2]We note that there are some trivial amplifications one can do by scaling the security parameter. For example, if we had a scheme that supported $n = \lambda$ keys, then for any constant c we could achieve KDM security for $n = \lambda^c$ keys. The transformation is to simply let $\lambda' = \lambda^c$ and then run the same scheme with security parameter λ'.

[3]While our construction does not need to know the 1-key circular secure scheme, it does need to know the specification of the CPA secure PKE.

projection functions (this is implied by 1-key circular security via [KM20]), denoted by $\mathsf{PRJ} = (\mathsf{PRJ.KeyGen}, \mathsf{PRJ.Enc}, \mathsf{PRJ.Dec})$, but we do not need to know this scheme for our construction.

Our key generation algorithm $\mathsf{KDM.KeyGen}(1^\lambda)$ begins by randomly sampling an "indicator value" $u \leftarrow \{0,1\}^\lambda$. Next it samples 2λ public secret key pairs as $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b}) \leftarrow \mathsf{CPA.KeyGen}(1^\lambda)$ for $i \in [\lambda]$, $b \in \{0,1\}$. The public key and secret key is set as $\overline{\mathsf{pk}} = \{\mathsf{pk}_{i,b}\}_{i\in[\lambda], b\in\{0,1\}}$, $\overline{\mathsf{sk}} = (u, \{sk_{i,u_i}\}_{i\in[\lambda]})$. Intuitively, all 2λ public keys are published, while only half of the secret keys as chosen by u are kept. This means that for all i any value which is later encrypted to pk_{i,u_i} can be recovered by the secret key and any value encrypted to $\mathsf{pk}_{i,1-u_i}$ cannot.

Our encryption algorithm $\mathsf{KDM.Enc}(\overline{\mathsf{pk}}, \mu)$ takes as input the public key and a message μ. It begins by first constructing a circuit C_μ that is a constant circuit of λ input bits that outputs μ on any input. (We pad it to an appropriate size, but leave details of padding to the main body description.) Next we created a garbling of $\mathcal{C}_\mu$ as $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda], b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\mu)$. Finally, we encrypt each label to the corresponding public key component as $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}_{i,b}, \mathsf{lab}_{i,b})$. And output the ciphertext as $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda], b\in\{0,1\}})$. The ciphertext consists of the garbled circuit $\widetilde{C}$ plus an encryption of each label.

The decryption algorithm utilizes the indicator value u to guide it to which labels it should attempt to recover. The algorithm recovers lab_{i,u_i} by decrypting ct_{i,u_i} with sk_{i,u_i} to for each i. At this point it has a label for each input and can recover the message by evaluating the garbled circuit. Since $\widetilde{C}$ is a garbling of the constant circuit C_μ it will output the message μ regardless of what set of label are recovered.

One can easily observe that our construction uses plain PKE and does not use any KDM scheme whatsoever. We also observe that our decryption process communicates the indicator value u to the garbled circuit. This doesn't matter much for the correctness of the construction since C_μ is a constant circuit; however, this feature will come into play further along in the proof.

Security for n-keys by key linking. We consider KDM security for n-keys where the attacker will get a single challenge ciphertext under the public key of the t-th user of some function f applied to the secret keys of all the users. The adversary chooses t, f, where the function f can be an arbitrary function of some a-prior bounded circuit size.[4] The general case of multiple challenge ciphertexts is handled almost as easily, but we restrict ourselves to a single one here for ease of exposition. Our proof proceeds over multiple steps.

In the first step we begin by linking the indicator values of these keys as follows. First, we introduce a 1-key KDM scheme PRJ for projection functions and sample $k \leftarrow \mathsf{PRJ.KeyGen}(1^\lambda)$. Without loss of generality, we can assume $|k| = \lambda$. Next choose random $v^j \leftarrow \{0,1\}^\lambda$ and set $u^j := k \oplus v^j$ for all $j \in [n]$ where u^j is the indicator component of the j-th key. This concludes the first step where we can now view the indicator components of the j-th key as being derived from a single k by XORing with v^j. Arguing indistinguishability of this step is immediate as it only involves a syntatic change and the distribution is the same as the prior hybrid. At this point the secret key of user j is only partly derived (in the indicator component) from k.

In the second step of the proof we let $\mathsf{ct}_{i,1-u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u_i^t}, 0^\lambda)\}_{i\in[\lambda]}$. That is replace the encryption of lab_{i,u_i^t} with the all 0's string, which effectively erases lab_{i,u_i^t} from the ciphertext. Indistinguishability of this step follows immediately from CPA security coupled with the fact that $\mathsf{sk}^t_{i,1-u_i^t}$ is *not* part of user t's secret key as it was discarded during key generation.

[4]The bound on the circuit size affects the size of the padding we need to add to the circuit C_μ in the construction.

In the third step, we now move to fully derive the all the secret keys from k when creating our ciphertext. To do this, we first encrypt all the secret keys $\overline{\mathsf{sk}}^j$ under k to derive ciphertexts $\widehat{\mathsf{ct}}^j \leftarrow \mathsf{PRJ.Enc}(k, \overline{\mathsf{sk}}^j)$ for $j \in [n]$. We then change from garbling the constant circuit C_μ to garbling a new circuit $\widehat{C}$ that has the ciphertext $\widehat{\mathsf{ct}}^j$ hard-coded and performs the following steps:

1. Takes in an input x, which should be u^t when decrypting properly with the t-th secret key.
2. Computes $x \oplus v^j$. Under the presumption $x = u^t$, this is equal to k.
3. Derives from k the secret keys $\overline{\mathsf{sk}}^j$ for all n users by decrypting $\overline{\mathsf{sk}}^j = \mathsf{PRJ.Dec}(k, \widehat{\mathsf{ct}}^j)$.[5]
4. Evaluate the function $f(\overline{\mathsf{sk}}^1, \ldots, \overline{\mathsf{sk}}^n)$ on the secret keys obtained from the previous step where f is the KDM query supplied by the attacker.

This proof step can be fulfilled by garbled circuit security. We can employ this argument since at this stage only one label per input bit is contained in the ciphertext.

At this point the keys are fully derived from the "master key" k. Before we can apply KDM security to erase the message we must first erase knowledge of k from the challenge ciphertext. Currently, k is indirectly embedded in the challenge ciphertext by the fact that labels are only encrypted in ct_{i,u_i^t} and not $\mathsf{ct}_{i,1-u_i^t}$. To remedy this we reverse an earlier step and add labels $\mathsf{lab}_{i,1-u_i^t}$ back to $\mathsf{ct}_{i,1-u_i^t}$.

With this modification the only information regarding k is inside the KDM ciphertexts $\widehat{\mathsf{ct}}^j$ contained within the garbled circuit. But now that k is gone elsewhere we can (finally) leverage KDM security to change $\widehat{\mathsf{ct}}^j$ to each be encryptions of the all 0's strings. Finally, now that all information regarding $\overline{\mathsf{sk}}^j$ is gone we can apply CPA security to remove *all* labels by changing both $\mathsf{ct}_{i,0}$ and $\mathsf{ct}_{i,1}$ to be encryptions of the all 0's string. And follow this by changing $\widetilde{C}$ to be a garbling of the constant function encrypting outputing the all 0's string.

1.2 Extensions and Comparisons

We conclude by mentioning a few extensions to our work as well as exploring technique comparisons.

Symmetric-Key KDM. The scheme described above shows how to go from 1-key KDM security to n-key KDM security for public key encryption, by additionally using standard CPA-secure PKE. We can also achieve the same results for secret key encryption without the additional assumption via a small modifications to our approach. One might try to achieve this by running the same system and simply replacing the use of a CPA secure public key encryption with CPA secure secret key encryption. This runs into the issue of how the encryption algorithm generates the values $\mathsf{ct}_{i,1-u_i}$ without a public key. One possibility is to add the components $\mathsf{sk}_{i,1-u_i}$ to the secret key; however, doing this will break our security argument. Instead, we will use a symmetric key encryption scheme with pseudorandom ciphertexts. (This is known from pseudorandom functions.) Then we can encrypt by setting $\mathsf{ct}_{i,1-u_i}$ to be a random string. The proof of security

[5] We actually only need to derive the secret keys j that are used in the computation of f. For a bounded size function f this may be considerably fewer than all n. However, for this overview we act as though all keys are derived here.

then proceeds in a mostly similar way to before. We give further details in the main body. Note that the resulting scheme depends on the symmetric-key CPA-secure encryption, but not on the 1-key KDM secure one – we only assume the latter exists, without knowing what it is.

In many applications it will be desirable to have KDM security with respect to an adversary that can also make chosen ciphertext attacks. As mentioned at the beginning starting with [KW19] several recent works have explored ways of achieving CCA security from key dependent type primitives. After applying our transformation for CPA security one can apply a theorem statement from [KM20] to then make the scheme CCA secure while maintaining KDM security for n-keys.

Given that prior schemes (e.g. [BHHI10, App11, KM20]) applied garbled circuits for function amplification, it is instructive to take a slightly closer look into the similarities and differences with our construction. We focus on the work of Applebaum [App11]. The construction of Applebaum is fairly straightforward. A user's public and secret key is created by calling the key generation algorithm for a scheme that is KDM secure for some weak functionality. To encrypt, one first simulates a garbled circuit that outputs the message μ — the simulation includes a simulated circuit and a single label for each input. These are all encrypted under the public key. To obtain the message the algorithm first decrypts the KDM ciphertext to get the garbled circuit and labels and then performs an evaluation which produces the message.

To prove security the challenge ciphertext is modified so that it is a garbling of a circuit which KDM function query f provided by the attacker and the labels given out correspond the the secret key of the underlying KDM scheme. The properties of the garbling construction imply that this garbling itself can be viewed as a projection function of the secret key. Security then follows from the fact that the underlying scheme was KDM with respect to projection functions.

One critical structural difference between our approach and the above is that in [App11] the starting KDM scheme is exposed on the outside. If we run an n-key KDM experiment the attacker will see all n keys. If the underlying scheme were only 1-key KDM secure, it is unclear how one would argue n-key security from an attacker that sees many 1-key secure schemes. In contrast our proof starts with a 1-key secure scheme, but it is hidden within the garbled circuit and not exposed to the outside. Moreover, the security proof only uses a single 1-KDM secure key. If there are multiple KDM ciphertexts, it will be under the *same* key over and over again. Interestingly, the structure of our key amplification is arguably closer to the structure used by [KMT19] in showing that hinting PRGs implied a encryption system that was one time KDM secure.

Another interesting comparison point is the construction of Kitagawa and Tanaka [KT18] for realizing KDM secure public key encryption from KDM secure secret key encryption and standard IND-CPA secure public key encryption. Their construction shares some stylistic features similar to ours in that λ pairs of public keys are generated and a symmetric key k is used to select which of each pair to keep in the secret key and which one to drop. Likewise, encryption will communicate a set of labels and the decryptor gets a set corresponding to the key k.

There are also some crucial differences in the approaches. First, in the Kitagawa and Tanaka scheme the evaluation of the garbled circuit during decryption produces a symmetric key KDM which must be then decrypted by k to get the message. In contrast our scheme never exposes the symmetric key ciphertext outside the garbled circuit during decryption. In fact (as mentioned before) no such scheme is actually used in the main construction and the value u is simply a random value and not a KDM secret key. This distinction allows us to instead link u to a single KDM key in our proof. And during the proof the symmetric key KDM ciphertexts are used within the garbled circuit giving us our universal property.

2 Preliminaries

For any integer $n \geq 1$, define $[n] = \{1, \ldots, n\}$. A function $\nu : \mathbb{N} \to \mathbb{N}$ is said to be negligible, denoted $\nu(n) = \mathrm{negl}(n)$, if for every positive polynomial $p(\cdot)$ and all sufficiently large n it holds that $\nu(n) < 1/p(n)$. We use the abbreviation PPT for probabilistic polynomial time. For a finite set S, we write $a \leftarrow S$ to mean a is sampled uniformly randomly from S. For a randomized algorithm A, we let $a \leftarrow A(\cdot)$ denote the process of running $A(\cdot)$ and assigning the outcome to a; when A is deterministic, we write $a := A(\cdot)$ instead. We denote the security parameter by λ. For two distributions X, Y parameterized by λ we say that they are computationally indistinguishable, denoted by $X \approx_c Y$ if for every PPT distinguisher D we have $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]| = \mathrm{negl}(\lambda)$.

2.1 Garbled Circuits

Definition 2.1 (Garbling Scheme). *A garbling scheme* $(\mathsf{Garble}, \mathsf{Eval})$ *has the following syntax:*

- $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C)$*: Takes as input a circuit C having n input wires and m output wires, and outputs the* garbled circuit $\widetilde{C}$ *along with* $2n$ labels $\mathsf{lab}_{i,b} \in \{0,1\}^\lambda$.
- $y = \mathsf{Eval}(\widetilde{C}, \{\mathsf{lab}_{i,x_i}\})$*: Given a garbled circuit $\widetilde{C}$ and n labels $\{\mathsf{lab}_{i,x_i}\}_{i\in[n]}$ for some input $x \in \{0,1\}^n$, outputs $y = C(x)$.*

Correctness: *For all circuits C with n input wires, for all $x \in \{0,1\}^n$, we have:*

$$\Pr[\mathsf{Eval}(\widetilde{C}, \{\mathsf{lab}_{i,x_i}\}_{i\in[n]}) = C(x)] = 1 - \mathrm{negl}(\lambda),$$

where the probability is over $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C)$.

Security: *There exists a PPT simulator* Sim *such that, for all circuits C with n input wires and m output wires, for all $x \in \{0,1\}^n$ we have*

$$(\widetilde{C}, \{\mathsf{lab}_{i,x_i}\}_{i\in[n]}) \approx_c \mathsf{Sim}(1^\lambda, 1^n, 1^m, 1^{|C|}, C(x))$$

where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C)$.

Claim 2.1.1 (Indistinguishability Security). *Assume* $(\mathsf{Garble}, \mathsf{Eval})$ *is a garbling scheme. Then for any circuits C_0, C_1 with n input wires, m output wires and equal circuit size $|C_0| = |C_1|$, for any input $x \in \{0,1\}^n$ such that $C_0(x) = C_1(x)$ and we have:*

$$(\widetilde{C}^0, \{\mathsf{lab}^0_{i,x_i}\}_{i\in[n]}) \approx_c (\widetilde{C}^1, \{\mathsf{lab}^1_{i,x_i}\}_{i\in[n]})$$

where $(\widetilde{C}^\beta, \{\mathsf{lab}^\beta_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\beta)$.

Proof. Follows via a hybrid argument. By the definition of garbled circuit security, there is a simulator Sim such that the following holds. Let $s := |C_0| = |C_1|$ and $y = C_0(x) = C_1(x)$. Then:

$$(\widetilde{C}^0, \{\mathsf{lab}^0_{i,x_i}\}_{i\in[n]}) \approx_c \mathsf{Sim}(1^\lambda, 1^n, 1^m, 1^s, y) \approx_c (\widetilde{C}^1, \{\mathsf{lab}^1_{i,x_i}\}_{i\in[n]}).$$

□

Claim 2.1.2 (No-Label Security)**.** *Assume* $(\mathsf{Garble}, \mathsf{Eval})$ *is a garbling scheme. Then for any circuits* C_0, C_1 *with* n *input wires,* m *output wires and equal circuit size* $|C_0| = |C_1|$ *we have*

$$\widetilde{C}^0 \approx_c \widetilde{C}^1$$

where $(\widetilde{C}^\beta, \{\mathsf{lab}^\beta_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\beta)$.

Proof. Let $y_0 = C_0(0^n)$ and $y_1 = C_1(0^n)$. Define the circuits

$$C_{alt(b)}(x) = \begin{cases} y_b & \text{if } x = 0^n \\ 0^m & \text{else} \end{cases}$$

and pad these circuits to be the same size as C_0, C_1.[6]

Let $(\widetilde{C}^\beta, \{\mathsf{lab}^\beta_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\beta)$ for $\beta \in \{0, 1, alt(0), alt(1)\}$.

Then

$$(\widetilde{C}^0, \{\mathsf{lab}^0_{i,0}\}_{i\in[n]}) \approx_c (\widetilde{C}^{alt(0)}, \{\mathsf{lab}^{alt(0)}_{i,0}\}_{i\in[n]})$$

by Claim 2.1.1 and the fact that $C_0(0^n) = C_{alt(0)}(0^n)$. Therefore $\widetilde{C}^0 \approx_c \widetilde{C}^{alt(0)}$.

Also,

$$(\widetilde{C}^{alt(0)}, \{\mathsf{lab}^{alt(0)}_{i,1}\}_{i\in[n]}) \approx_c (\widetilde{C}^{alt(1)}, \{\mathsf{lab}^{alt(1)}_{i,1}\}_{i\in[n]})$$

by Claim 2.1.1 and the fact that $C_{alt(0)}(1^n) = C_{alt(1)}(1^n) = 0^m$. Therefore $\widetilde{C}^{alt(0)} \approx_c \widetilde{C}^{alt(1)}$.

Lastly,

$$(\widetilde{C}^{alt(1)}, \{\mathsf{lab}^{alt(1)}_{i,0}\}_{i\in[n]}) \approx_c (\widetilde{C}^1, \{\mathsf{lab}^1_{i,0}\}_{i\in[n]})$$

by Claim 2.1.1 and the fact that $C_{alt(1)}(0^n) = C_1(1^n)$. Therefore $\widetilde{C}^{alt(1)} \approx_c \widetilde{C}^1$.

By a hybrid argument, we therefore get $\widetilde{C}^0 \approx_c \widetilde{C}^1$. $\square$

2.2 KDM Security Definitions

Definition 2.2 (KDM Security [BRS03])**.** *Let* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a public-key encryption scheme with the usual correctness property. Let* $n = n(\lambda)$ *be some polynomial and let* $\mathcal{F} = \{\mathcal{F}_\lambda\}$ *be some family of functions. We say that the scheme is* $(n, \mathcal{F})$*-KDM if for all PPT adversary* $\mathcal{A}$ *we have*

$$|\Pr[\mathsf{KDMGame}^0_\mathcal{A}(1^\lambda) = 1] - \Pr[\mathsf{KDMGame}^1_\mathcal{A}(1^\lambda)]| = \mathrm{negl}(\lambda)$$

where $\mathsf{KDMGame}^\beta_\mathcal{A}(1^\lambda)$ *is defined as the output of* $\mathcal{A}$ *in the following game:*

- *For* $j \in [n]$*, the challenger chooses* $(\mathsf{pk}^j, \mathsf{sk}^j) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and gives* $\mathsf{pk}^1, \ldots, \mathsf{pk}^n$ *to* $\mathcal{A}$*.*
- *The adversary* $\mathcal{A}$ *makes arbitrary many KDM encryption queries:*
  - *The adversary* $\mathcal{A}$ *chooses a pair* (t, f) *where* $t \in [n]$ *and* $f \in \mathcal{F}_\lambda$*.*
  - *The challenger replies with* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}^t, \mu)$ *were* μ *is defined as:*
    * *If* $\beta = 0$*, then* $\mu := 0^\ell$ *where* ℓ *is the output length of* f*.*
    * *If* $\beta = 1$ *then* $\mu := f(\mathsf{sk}^1, \ldots, \mathsf{sk}^n)$*.*

[6]We assume that $|C_b|$ is at least as large as $C_{alt(b)}$, which is of some fixed size $O(n + m)$. We can make this hold without loss of generality by always padding all circuits to be at least of that size before garbling them.

By default, we only consider bit-encryption schemes whose native message space is 1-bit. However, we will abuse notation and write $\mathsf{Enc}(\mathsf{sk}, \mu)$ for longer messages μ to denote encrypting μ bitwise, one bit at a time. Also, we always allow the adversary to choose the constant-0 or constant-1 functions, which corresponds to encrypting 0 or 1, even when we do not explicitly define these as part of the class $\mathcal{F}$. Lastly, we note that the definition of $(n, \mathcal{F})$ security only makes sense syntactically when $\mathcal{F}_\lambda$ contains functions f whose inputs consist of $n(\lambda)$ secret keys. We specify some classes of interest below.

(n, s)-circuit-KDM. For a polynomial $s = s(\lambda)$, we say that an encryption scheme is (n, s)-circuit-KDM secure if it is $(n, \mathcal{F})$-KDM secure, where $\mathcal{F} = \{\mathcal{F}_\lambda\}$ is the class of circuits having size $s(\lambda)$. Although we think of functions $f \in \mathcal{F}$ as taking the input $\mathsf{sk}^1, \ldots, \mathsf{sk}^n$, it will be meaningful to talk about (n, s)-KDM security for $s \ll n$. In this case, the function's circuit size is too small for it to even read the entire input. Instead, we think of the circuit f as specifying, for each input wire, a pair of indices (i, j) indicating that this input wire reads the j'th bit of the i'th secret key sk^i. We let $\mathsf{inp}(f) \subseteq [n]$ denote the subset of the secret keys that the input wires of f depend on, and we can write $f(\{\mathsf{sk}^i\}_{i \in \mathsf{inp}(f)})$ in place of $f(\mathsf{sk}^1, \ldots, \mathsf{sk}^n)$.

n-circular. We say that a scheme is *n-circular secure* if it is $(n, \mathcal{F})$-KDM secure where $\mathcal{F}$ consists of the class of functions $f_{i,j}(\mathsf{sk}^1, \ldots, \mathsf{sk}^n)$ that output the j'th bit of the i'th secret key sk^i for some (i, j).

n-projection-KDM. We say that a scheme is n-projection-KDM secure if it is $(n, \mathcal{F})$-KDM secure where $\mathcal{F}$ consists of the class of functions $f_{i,j,b}(\mathsf{sk}^1, \ldots, \mathsf{sk}^n) = f_{i,j}(\mathsf{sk}^1, \ldots, \mathsf{sk}^n) \oplus b$ for arbitrary (i, j, b) and for $f_{i,j}$ as defined above. When extended to multi-bit messages encrypted bit-wise, n-projection-KDM implies security for the class of functions $\mathcal{F}$ where each output bit can depend arbitrarily on any single bit of $\mathsf{sk}^1, \ldots, \mathsf{sk}^n$.

$(\infty, \mathcal{F})$-KDM. We say that a scheme is $(\infty, \mathcal{F})$-KDM secure if it $(n, \mathcal{F})$-KDM secure for all polynomial $n = n(\lambda)$. The notions of (∞, s)-circuit-KDM security and ∞-circular-security are defined analogously.

Symmetric-Key KDM. We define KDM security for symmetric-key encryption analogously to the above definition for public key encryption. In particular, $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$ now only outputs a secret key (and no public key), and $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \mu)$ now takes a secret key as an input instead of the public key. Security is defined analogously by modifying the KDM security so that (i) the adversary does not get any $\mathsf{pk}^1, \ldots, \mathsf{pk}^n$, (ii) the KDM encryption queries are now answered via a symmetric-key encryption $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}^t, u)$.

3 Amplifying KDM

Let $\mathsf{CPA} = (\mathsf{CPA.KeyGen}, \mathsf{CPA.Enc}, \mathsf{CPA.Dec})$ be a CPA-secure public-key encryption scheme. Let $(\mathsf{Garble}, \mathsf{Eval})$ be a garbled circuit scheme (Definition 2.1). We construct a public-key encryption scheme $\mathsf{KDM} = (\mathsf{KDM.KeyGen}, \mathsf{KDM.Enc}, \mathsf{KDM.Dec})$, parameterized by some padding parameter $p = p(\lambda)$ to be specified later, as follows:

$(\overline{\mathsf{pk}}, \overline{\mathsf{sk}}) \leftarrow \mathsf{KDM.KeyGen}(1^\lambda)$**:** For $i \in [\lambda]$, $b \in \{0,1\}$, sample $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b}) \leftarrow \mathsf{CPA.KeyGen}(1^\lambda)$. Sample $u \leftarrow \{0,1\}^\lambda$. Let $\overline{\mathsf{pk}} = \{\mathsf{pk}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}$, $\overline{\mathsf{sk}} = (u, \{sk_{i,u_i}\}_{i\in[\lambda]})$.

$\overline{\mathsf{ct}} \leftarrow \mathsf{KDM.Enc}(\overline{\mathsf{pk}}, \mu)$**:** Let C_μ be the constant circuit that takes any input $x \in \{0,1\}^\lambda$ and always outputs μ. We pad C_μ to be of size $p(\lambda)$ for the padding parameter p. Let

$$(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\mu).$$

For $i \in [\lambda], b \in \{0,1\}$, sample $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}_{i,b}, \mathsf{lab}_{i,b})$. Output $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$.

$\mu := \mathsf{KDM.Dec}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}})$**:** For $i \in [\lambda]$, let $\mathsf{lab}_{i,u_i} := \mathsf{CPA.Dec}(\mathsf{sk}_{i,u_i}, \mathsf{ct}_{i,u_i})$. Output $\mu := \mathsf{Eval}(\widetilde{C}, \{\mathsf{lab}_{i,u_i}\})$.

Correctness Let $\overline{\mathsf{pk}} = \{\mathsf{pk}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}$, $\overline{\mathsf{sk}} = (u, \{sk_{i,u_i}\}_{i\in[\lambda]})$ be a key pair output by the KDM.KeyGen algorithm. And let $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ be a ciphertext produced from $\mathsf{KDM.Enc}(\overline{\mathsf{pk}}, \mu)$ for some μ. When the algorithm $\mathsf{KDM.Dec}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}})$ is run it first produces $\mathsf{lab}_{i,u_i} := \mathsf{CPA.Dec}(\mathsf{sk}_{i,u_i}, \mathsf{ct}_{i,u_i})$. By the correctness of the underlying CPA secure encryption scheme $\mathsf{lab}_{i,u_i} = \mathsf{lab}'_{i,u_i}$ for all i where lab'_{i,u_i} are the labels generated during the encryption procedure. Next, we have that $\mathsf{Eval}(\widetilde{C}, \{\mathsf{lab}_{i,u_i}\}) = C_\mu(u)$ by the correctness of the garbling scheme where μ is the message encrypted. Finally, $C_\mu(u) = \mu$ since C_μ is the constant function and correctness holds.

3.1 Security

Theorem 3.1. *Let* CPA *be a CPA-secure public-key encryption scheme and let* (Garble, Eval) *be a secure garbling scheme. Then for any polynomial $s(\lambda)$ there exists some polynomial padding parameter $p(\lambda)$ such that the following holds. Assume that there exists some 1-circular symmetric-key encryption scheme. Then the above* KDM *scheme with padding parameter p is an $(\infty, s(\lambda))$-circuit-KDM secure public-key encryption.*

Proof. We define the polynomial $p(\lambda)$ further below. Assume there exists some symmetric-key 1-circular secure bit-encryption scheme. Then by [KM20] there also exists 1-projection-KDM secure symmetric-key encryption denoted by PRJ = (PRJ.KeyGen, PRJ.Enc, PRJ.Dec).

By rescaling the security parameter, we can assume without loss of generality that: (i) the secret key size of $k \leftarrow \mathsf{PRJ.KeyGen}(1^\lambda)$ is of size $|k| \le \lambda$ and (ii) the circuit size of the decryption function $\mathsf{PRJ.Dec}(\cdot, \mathsf{ct})$ (which has a hard-coded bit-ciphertext ct, takes as input the key k, and outputs $\mathsf{PRJ.Dec}(k, \mathsf{ct})$) is bounded by $|\mathsf{PRJ.Dec}(\cdot, \mathsf{ct})| \le \lambda$. In particular, assume we have some scheme PRJ′ where the maximum of the keys size $|k|$ and the circuit size of PRJ′.Dec are bounded by some potentially large polynomial $q(\lambda) \le c_1 \cdot \lambda^{c_2}$ for some constants c_1, c_2. We can define the scheme PRJ by taking $\mathsf{PRJ.KeyGen}(1^\lambda) = \mathsf{PRJ'.KeyGen}(1^{\lambda'})$ where $\lambda' = \lambda^{1/c_2}/c_1$ to satisfy (ii) and ensure $|k| \le \lambda$. It is easy to see that asymptotic security is preserved, although exact security degrades. Furthermore, we can strengthen (i) to require that the key size is exactly $|k| = \lambda$ while preserving (ii). We do this by padding the key with dummy bits, which the decryption circuit ignores (this allows us to have a potentially smaller decryption circuit than key size).

For any polynomial $n = n(\lambda)$, we prove the (n, s)-circuit-KDM security of the KDM scheme via the followng sequence of hybrids:

Hybrid 0. This is (n, s)-KDM security game of the scheme KDM with the challenge bit $\beta = 1$, which proceeds as follows:

- For $j \in [n]$, the challenger chooses $(\overline{\mathsf{pk}}^j, \overline{\mathsf{sk}}^j) \leftarrow \mathsf{KDM.KeyGen}(1^\lambda)$ and gives $\overline{\mathsf{pk}}^1, \ldots, \overline{\mathsf{pk}}^n$ to $\mathcal{A}$.
- The adversary $\mathcal{A}$ makes arbitrary many KDM encryption queries:
  - The adversary $\mathcal{A}$ chooses a pair (t, f) where $t \in [n]$ and f is a circuit of size $|f| \leq s$.
  - The challenger sets $\mu := f(\{\overline{\mathsf{sk}}^j\}_{j \in \mathsf{inp}(f)})$ and replies with $\overline{\mathsf{ct}} \leftarrow \mathsf{KDM.Enc}(\overline{\mathsf{pk}}^t, \mu)$.

Hybrid 1. We modify how the values $(\overline{\mathsf{pk}}^j, \overline{\mathsf{sk}}^j)$ are chosen by the challenger.

In Hybrid 0, the challenger chooses $(\overline{\mathsf{pk}}^j, \overline{\mathsf{sk}}^j) \leftarrow \mathsf{KDM.KeyGen}(1^\lambda)$ as follows:

- Sample $u^j \leftarrow \{0,1\}^\lambda$.
- Sample $(\mathsf{pk}^j_{i,b}, \mathsf{sk}^j_{i,b}) \leftarrow \mathsf{CPA.KeyGen}(1^\lambda)$. Let $\overline{\mathsf{pk}}^j := \{\mathsf{pk}^j_{i,b}\}_{i \in [\lambda], b \in \{0,1\}}$, $\overline{\mathsf{sk}} := (u^j, \{sk^j_{i,u^j_i}\}_{i \in [\lambda]})$.

In Hybrid 1, the challenger first samples $k \leftarrow \mathsf{PRJ.KeyGen}(1^\lambda)$. The challenger sets $(\overline{\mathsf{pk}}^j, \overline{\mathsf{sk}}^j)$ as:

- Sample $v^j \leftarrow \{0,1\}^\lambda$ and set $u^j := k \oplus v^j$.
- Sample $(\mathsf{pk}^j_{i,b}, \mathsf{sk}^j_{i,b}) \leftarrow \mathsf{CPA.KeyGen}(1^\lambda)$. Let $\overline{\mathsf{pk}}^j := \{\mathsf{pk}^j_{i,b}\}_{i \in [\lambda], b \in \{0,1\}}$, $\overline{\mathsf{sk}}^j := (u^j, \{sk^j_{i,u^j_i}\}_{i \in [\lambda]})$.

The challenger uses the PRJ scheme to encrypt all the secret keys: $\widehat{\mathsf{ct}}^j \leftarrow \mathsf{PRJ.Enc}(k, \overline{\mathsf{sk}}^j)$ for $j \in [n]$. It stores the ciphertexts $\widehat{\mathsf{ct}}^j$, but does not do anything with them yet in this hybrid.

It is easy to see that the above only introduces a syntactical difference, but the distribution of hybrids 0 and 1 is identical since $u^j = k \oplus v^j$ is uniform over the choice of v^j. Note that the ciphertexts $\widehat{\mathsf{ct}}^j$ do not appear in the view of the adversary.

Hybrid 2. We now modify how KDM encryption queries (t, f) are answered.

In Hybrid 1, the challenger chooses $\overline{\mathsf{ct}} \leftarrow \mathsf{KDM.Enc}(\overline{\mathsf{pk}}^t, \mu)$, where $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i \in [\lambda], b \in \{0,1\}})$ with $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i \in [\lambda], b \in \{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\mu)$, and $\{\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, \mathsf{lab}_{i,b})\}_{i \in [\lambda], b \in \{0,1\}}$.

In Hybrid 2, the challenger replaces $\mathsf{ct}_{i,b}$ with a dummy encryption of 0's when $b = 1 - u^t_i$. That is, the challenger chooses $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i \in [\lambda], b \in \{0,1\}})$ with $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i \in [\lambda], b \in \{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\mu)$, and $\{\mathsf{ct}_{i,u^t_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,u^t_i}, \mathsf{lab}_{i,u^t_i})\ ,\ \mathsf{ct}_{i,1-u^t_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u^t_i}, 0^\lambda)\}_{i \in [\lambda]}$.

These hybrids are computationally indistinguishable by the CPA security of the encryption scheme CPA*. In particular, the secret keys $\{sk^t_{i,1-u^t_i}\}_{i \in [\lambda]}$ are not used anywhere in the game and are not a part of $\overline{\mathsf{sk}}^t$. Therefore, CPA security allows us to replace $\mathsf{ct}_{i,1-u^t_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u^t_i}, \mathsf{lab}_{i,1-u^t_i})$ by $\mathsf{ct}_{i,1-u^t_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u^t_i}, 0^\lambda)$. Formally, proving indistinguishability of Hybrid 1 and 2 requires an internal hybrid argument over all the encryption queries and all the indices $i \in [\lambda]$, where we replace each such ciphertext $\mathsf{ct}_{i,1-u^t_i}$ one by one.*

Hybrid 3. We modify how the KDM encryption queries (t, f) are answered once more. This time we change how the garbled circuit is sampled.

In Hybrid 2, we set $\overline{\mathsf{ct}} := (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\mu)$, and $\{\mathsf{ct}_{i,u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,u_i^t}, \mathsf{lab}_{i,u_i^t})\ ,\ \mathsf{ct}_{i,1-u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u_i^t}, 0^\lambda)\}_{i\in[\lambda]}$.

In Hybrid 3, for each KDM encryption query (t, f) the challenger constructs a circuit $\widehat{C}$ that contains hard-coded values $f, v^t, \{\widehat{\mathsf{ct}}^j\}_{j\in[\mathsf{inp}(f)]}$ and is defined as follows:

$\widehat{C}(x)$ {

- Compute $k' := v^t \oplus x$.
- For $j \in \mathsf{inp}(f)$: compute $\overline{\mathsf{sk}}^j \leftarrow \mathsf{PRJ.Dec}(k', \widehat{\mathsf{ct}}^j)$.
- Output $f(\{\overline{\mathsf{sk}}^j\}_{j\in\mathsf{inp}(f)})$

}

The challenger sets $\overline{\mathsf{ct}} := (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$, where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, \widehat{C})$, and

$$\{\mathsf{ct}_{i,u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,u_i^t}, \mathsf{lab}_{i,u_i^t})\ ,\ \mathsf{ct}_{i,1-u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u_i^t}, 0^\lambda)\}_{i\in[\lambda]}.$$

We also now define the padding parameter $p(\lambda)$ to be the circuit size of the circuit $\widehat{C}$. Note that $p(\lambda)$ only depends on $s(\lambda)$ but is independent of n. It is also independent of the exact parameters of the 1-circular secure bit-encryption, since no matter which such encryption we started with, we ensured the PRJ decryption circuit has size $\leq \lambda$ to decrypt each bit.

These hybrids are computationally indistinguishable by the indistinguishability security of garbled circuits (see Claim 2.1.1). In particular, recall that $v^t = u^t \oplus k$ and therefore, on input $x = u^t$ we have $k' = k$ during the computation of $\widehat{C}(x)$ and hence $\widehat{C}(u^t) = f(\{\overline{\mathsf{sk}}^j\}_{j\in\mathsf{inp}(f)}) = C_\mu(u^t)$. Furthermore, in both hybrids, the adversary's view only depends on the garbled circuit $\widetilde{C}$ and the labels $\{\mathsf{lab}_{i,u_i^t}\}_{i\in[\lambda]}$, but not on the other labels $\{\mathsf{lab}_{i,1-u_i^t}\}_{i\in[\lambda]}$. Therefore, we can rely on indistinguishability security of garbled circuits to replace the circuit being garbled from C_μ to $\widehat{C}$. Formally, proving indistinguishability of Hybrids 2 and 3 requires an internal hybrid argument over all the KDM encryption queries, where we replace the circuit being garbled in each query one by one.

Hybrid 4. We modify how the KDM encryption queries are answered once again.

In Hybrid 3, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, \widehat{C})$ and $\left\{\mathsf{ct}_{i,u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,u_i^t}, \mathsf{lab}_{i,u_i^t}), \mathsf{ct}_{i,1-u_i^t} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u_i^t}, 0^\lambda)\right\}_{i\in[\lambda]}$.

In Hybrid 4, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, \widehat{C})$ and $\left\{\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, \mathsf{lab}_{i,b})\right\}_{i\in[\lambda],b\in\{0,1\}}$.

These hybrids are computationally indistinguishable by the CPA security of the encryption scheme CPA. *In particular, the secret keys $\{sk^t_{i,1-u^t_i}\}_{i\in[\lambda]}$ are not used anywhere in the game and are not a part of $\overline{\mathsf{sk}}^t$. Therefore, we can replace $\mathsf{ct}_{i,1-u^t_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u^t_i}, 0^\lambda)$ by $\mathsf{ct}_{i,1-u^t_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,1-u^t_i}, \mathsf{lab}_{i,1-u^t_i})$. Formally, proving indistinguishability of Hybrid 3 and 4 requires an internal hybrid argument over all the encryption queries and all the indices $i \in [\lambda]$, where we replace how we sample each such ciphertext $\mathsf{ct}_{i,1-u^t_i}$ one by one.*

Hybrid 5. We now change how the ciphertexts $\widehat{\mathsf{ct}}^j$ are chosen during key generation. Recall that these ciphertexts are hard-coded in the circuits $\widehat{C}$ used to answer KDM encryption queries.

In Hybrid 4, the challenger chooses $\widehat{\mathsf{ct}}^j \leftarrow \mathsf{PRJ.Enc}(k, \overline{\mathsf{sk}}^j)$ for $j \in [n]$.

In Hybrid 5, the challenger chooses $\widehat{\mathsf{ct}}^j \leftarrow \mathsf{PRJ.Enc}(k, 0^\alpha)$ for $j \in [n]$, where $\alpha = \left|\overline{\mathsf{sk}}^j\right|$.

These hybrids are computationally indistinguishable by the 1-projection-KDM security of the scheme PRJ. *To see this at a high level, note that we can think of $\overline{\mathsf{sk}}^j = \{\mathsf{sk}^j_{i,u^j_i}\}$ as being defined by the function $\overline{\mathsf{sk}}^j = \phi^j(k)$, where ϕ^j knows the values v^j and $\{\mathsf{sk}^j_{i,b}\}$, computes $u^j = k \oplus v^j$ and sets $\overline{\mathsf{sk}}^j = \{\mathsf{sk}^j_{i,u^j_i}\}$; each bit of $\overline{\mathsf{sk}}^j$ depends on a single bit of u^j, which in tun depends on a single bit of k, making ϕ^j a projection function.*

Formally, we give a reduction showing the indistinguishability of Hybrid 4 and 5. The reduction samples:

- $v^j \leftarrow \{0,1\}^\lambda$ *for* $j \in [n]$.
- $(\mathsf{pk}^j_{i,b}, \mathsf{sk}^j_{i,b}) \leftarrow \mathsf{CPA.KeyGen}(1^\lambda)$ *for* $j \in [n], i \in [\lambda], b \in \{0,1\}$. *Let* $\overline{\mathsf{pk}}^j := \{\mathsf{pk}^j_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}$.

Define the functions $\phi^j(k)$ where ϕ^j has the values v^j and $\{\mathsf{sk}^j_{i,b}\}$ hard-coded, computes $u^j = k \oplus v^j$ and sets $\overline{\mathsf{sk}}^j = \{\mathsf{sk}^j_{i,u^j_i}\}$. This is a projection function. The reduction makes encryption queries to the PRJ *scheme with the function ϕ^j and gets back ciphertexts $\widehat{\mathsf{ct}}^j$ for $j \in [n]$. The reduction then runs the adversary $\mathcal{A}$ by giving it the public keys $\overline{\mathsf{pk}}^j$ and answering KDM encryption queries exactly as the challenger in Hybrid 4 using the values $v^j, \{\widehat{\mathsf{ct}}^j\}$. It outputs whatever $\mathcal{A}$ outputs at the end.*

If the reduction received ciphertexts $\widehat{\mathsf{ct}}^j \leftarrow \mathsf{PRJ.Enc}(k, \phi^j(k))$ then the above matches the distribution of Hybrid 4, and if the reduction received ciphertexts $\widehat{\mathsf{ct}}^j \leftarrow \mathsf{PRJ.Enc}(k, 0^\alpha)$ then the above matches the distribution of Hybrid 5. Therefore, the reduction break the projection-KDM security of the PRJ *scheme with the same advantage as that of $\mathcal{A}$ in distinguishing Hybrids 4 and 5.*

Hybrid 6. We modify how the KDM encryption queries (t, f) are answered.

In Hybrid 5, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, \widehat{C})$ and $\left\{\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, \mathsf{lab}_{i,b})\right\}_{i\in[\lambda],b\in\{0,1\}}$.

In Hybrid 6, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, \widehat{C})$ and $\left\{\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, 0^\lambda))\right\}_{i\in[\lambda],b\in\{0,1\}}$.

These hybrids are computationally indistinguishable by the CPA security of the encryption scheme CPA. *In particular, now the secret keys* $\{\mathsf{sk}^t_{i,b}\}$ *are not used anywhere in the game after they are generated! Formally, proving indistinguishability of Hybrid 5 and 6 requires an internal hybrid argument over all the encryption queries and all the indices* $i \in [\lambda], b \in \{0, 1\}$, *where we replace how we sample each such ciphertext* $\mathsf{ct}_{i,b}$ *one by one.*

Hybrid 7. We modify how the KDM encryption queries are answered again.

In Hybrid 6, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, \widehat{C})$ and $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, 0^\lambda)$.

In Hybrid 7, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_{0^\ell})$, for ℓ being the output size of f, and $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, 0^\lambda)$. The circuit $C_{0^\ell}(x)$ always outputs the constant 0^ℓ on every input x, but we pad the circuit to be of size $p(\lambda)$.

These hybrids are computationally indistinguishable by the garbled circuit security with no labels (Claim 2.1.2). In particular, the labels $\mathsf{lab}_{i,b}$ *are not used anywhere else in game after they are generated. Formally, proving indistinguishability of Hybrids 6 and 7 requires an internal hybrid argument over all the KDM encryption queries, where we replace the circuit being garbled in each query one by one.*

Hybrid 8. We modify how the KDM encryption queries are answered once more.

In Hybrid 7, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_{0^\ell})$, for ℓ being the output size of f, and $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, 0^\lambda)$.

In Hybrid 8, we set $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$ where $(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_{0^\ell})$, for ℓ being the output size of f, and $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, \mathsf{lab}_{i,b})$.

These hybrids are computationally indistinguishable by the CPA security of the encryption scheme CPA. *In particular, the secret keys* $\{\mathsf{sk}^t_{i,b}\}$ *are not used anywhere in the game after they are generated.*

Formally, proving indistinguishability of Hybrids 7 and 8 requires an internal hybrid argument over all the encryption queries and all the indices $i \in [\lambda], b \in \{0,1\}$, where we replace how we sample each such ciphertext $\mathsf{ct}_{i,b}$ one by one.

Note that Hybrid 8 is equivalent to the (n,s)-KDM security game of the scheme KDM with the challenge bit $\beta = 0$, which proceeds as follows:

- For $j \in [n]$, the challenger chooses $(\overline{\mathsf{pk}}^j, \overline{\mathsf{sk}}^j) \leftarrow \mathsf{KDM.KeyGen}(1^\lambda)$ and gives $\overline{\mathsf{pk}}^1, \ldots, \overline{\mathsf{pk}}^n$ to $\mathcal{A}$.
- The adversary $\mathcal{A}$ makes arbitrary many KDM encryption queries:
  - The adversary $\mathcal{A}$ chooses a pair (t, f) where $t \in [n]$ and f is a circuit of size $|f| \le s$.
  - The challenger set $\mu = 0^\ell$ where ℓ is the output size of f and replies with $\overline{\mathsf{ct}} \leftarrow \mathsf{KDM.Enc}(\overline{\mathsf{pk}}^t, \mu)$.

Therefore, the indistinguishability of Hybrid 0 and Hybrid 8 as shows above implies the security of the scheme KDM as claimed.

□

3.2 Symmetric-Key Setting

We show the same amplification from 1-circular to KDM security also holds in the symmetric-key setting. Let CPA = (CPA.KeyGen, CPA.Enc, CPA.Dec) be a CPA-secure symmetric-key encryption scheme with pseudorandom ciphertexts and ciphertext space $\mathcal{C}_\lambda$. Let (Garble, Eval) be a garbled circuit scheme (Definition 2.1). We construct a symmetric-key encryption scheme KDM = (KDM.KeyGen, KDM.Enc, KDM.Dec), parameterized by some padding parameter $p = p(\lambda)$ to be specified later, as follows:

$\overline{\mathsf{sk}} \leftarrow \mathsf{KDM.KeyGen}(1^\lambda)$**:** For $i \in [\lambda]$, $b \in \{0,1\}$, sample $\mathsf{sk}_{i,b} \leftarrow \mathsf{CPA.KeyGen}(1^\lambda)$. Sample $u \leftarrow \{0,1\}^\lambda$. Let $\overline{\mathsf{sk}} = (u, \{sk_{i,u_i}\}_{i\in[\lambda]})$.

$\overline{\mathsf{ct}} \leftarrow \mathsf{KDM.Enc}(\overline{\mathsf{sk}}, \mu)$**:** Let C_μ be the constant circuit that takes any input $x \in \{0,1\}^\lambda$ and always outputs μ. We pad C_μ to be of size $p(\lambda)$ for the padding parameter p. Let

$$(\widetilde{C}, \{\mathsf{lab}_{i,b}\}_{i\in[\lambda], b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C_\mu).$$

For $i \in [\lambda]$:

- Sample $\mathsf{ct}_{i,u_i} \leftarrow \mathsf{CPA.Enc}(\mathsf{sk}_{i,u_i}, \mathsf{lab}_{i,u_i})$.
- Sample $\mathsf{ct}_{i,1-u_i} \leftarrow \mathcal{C}_\lambda$.

Output $\overline{\mathsf{ct}} = (\widetilde{C}, \{\mathsf{ct}_{i,b}\}_{i\in[\lambda], b\in\{0,1\}})$.

$\mu := \mathsf{KDM.Dec}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}})$**:** For $i \in [\lambda]$, let $\mathsf{lab}_{i,u_i} := \mathsf{CPA.Dec}(\mathsf{sk}_{i,u_i}, \mathsf{ct}_{i,u_i})$. Output $\mu := \mathsf{Eval}(\widetilde{C}, \{\mathsf{lab}_{i,u_i}\})$.

Theorem 3.2. *Let* CPA *be a CPA-secure symmetric-key encryption scheme with pseudorandom ciphertexts and let* (Garble, Eval) *be a secure garbling scheme, both of which follow from one-way function. Then for any polynomial $s(\lambda)$ there exists some polynomial padding parameter $p(\lambda)$ such that the following holds. Assume that there exists some 1-circular symmetric-key encryption scheme. Then the above* KDM *scheme with padding parameter p is an $(\infty, s(\lambda))$-circuit-KDM secure secret-key encryption scheme.*

The proof of correctness and security are essentially the same as in the public-key scheme, with the following minor changes. Firstly, we can skip hybrid 2 since, already in the actual scheme, we choose $\mathsf{ct}_{i,1-u_i} \leftarrow \mathcal{C}_\lambda$ as dummy ciphertexts that don't contain any information about the label lab_{i,u_i}. All other hybrids are defined analogously with all occurrences of $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, \mathsf{lab}_{i,b})$ replaced by $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{sk}^t_{i,b}, \mathsf{lab}_{i,b})$ and all occurrences of $\mathsf{ct}_{i,b} \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}^t_{i,b}, 0^\ell)$ replaced by $\mathsf{ct}_{i,b} \leftarrow \mathcal{C}_\lambda$. Lastly, we can again skip hybrid 8 for the same reason as why we skipped hybrid 2.

References

[ABBC10] Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash. Cryptographic agility and its relation to circular encryption. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 403–422. Springer, Heidelberg, May / June 2010. 2

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, Heidelberg, August 2009. 2

[AP16] Navid Alamati and Chris Peikert. Three's compromised too: Circular insecurity for any cycle length from (ring-)LWE. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 659–680. Springer, Heidelberg, August 2016. 2

[App11] Benny Applebaum. Key-dependent message security: Generic amplification and completeness. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 527–546. Springer, Heidelberg, May 2011. 2, 3, 6

[BGK11] Zvika Brakerski, Shafi Goldwasser, and Yael Tauman Kalai. Black-box circular-secure encryption beyond affine functions. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 201–218. Springer, Heidelberg, March 2011. 2

[BHHI10] Boaz Barak, Iftach Haitner, Dennis Hofheinz, and Yuval Ishai. Bounded key-dependent message security. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 423–444. Springer, Heidelberg, May / June 2010. 2, 3, 6

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, Heidelberg, August 2008. 2, 3

[BHW15] Allison Bishop, Susan Hohenberger, and Brent Waters. New circular security counterexamples from decision linear and learning with errors. In Tetsu Iwata and

Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 776–800. Springer, Heidelberg, November / December 2015. 2

[BLSV18] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 535–564. Springer, Heidelberg, April / May 2018. 2, 3

[BRS03] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *SAC 2002: 9th Annual International Workshop on Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75. Springer, Heidelberg, August 2003. 2, 8

[CDG$^+$17] Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 33–65. Springer, Heidelberg, August 2017. 2

[CGH12] David Cash, Matthew Green, and Susan Hohenberger. New definitions and separations for circular security. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012: 15th International Conference on Theory and Practice of Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 540–557. Springer, Heidelberg, May 2012. 2

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, Heidelberg, May 2001. 1

[DG17] Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 537–569. Springer, Heidelberg, August 2017. 2

[GKW17a] Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In Chris Umans, editor, *58th Annual Symposium on Foundations of Computer Science*, pages 612–621. IEEE Computer Society Press, October 2017. 2

[GKW17b] Rishab Goyal, Venkata Koppula, and Brent Waters. Separating IND-CPA and circular security for unbounded length key cycles. In Serge Fehr, editor, *PKC 2017: 20th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10174 of *Lecture Notes in Computer Science*, pages 232–246. Springer, Heidelberg, March 2017. 2

[GKW17c] Rishab Goyal, Venkata Koppula, and Brent Waters. Separating semantic and circular security for symmetric-key bit encryption from the learning with errors assumption. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 528–557. Springer, Heidelberg, April / May 2017. 2

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th Annual ACM Symposium on Theory of Computing*, pages 365–377. ACM Press, May 1982. 2

[HK15] Mohammad Hajiabadi and Bruce M. Kapron. Reproducible circularly-secure bit encryption: Applications and realizations. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 224–243. Springer, Heidelberg, August 2015. 2

[KM19] Fuyuki Kitagawa and Takahiro Matsuda. CPA-to-CCA transformation for KDM security. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 118–148. Springer, Heidelberg, December 2019. 2

[KM20] Fuyuki Kitagawa and Takahiro Matsuda. Circular security is complete for KDM security. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 253–285. Springer, Heidelberg, December 2020. 2, 3, 4, 6, 10

[KMT19] Fuyuki Kitagawa, Takahiro Matsuda, and Keisuke Tanaka. CCA security and trapdoor functions via key-dependent-message security. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 33–64. Springer, Heidelberg, August 2019. 2, 6

[KRW15] Venkata Koppula, Kim Ramchen, and Brent Waters. Separations in circular security for arbitrary length key cycles. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 378–400. Springer, Heidelberg, March 2015. 2

[KT18] Fuyuki Kitagawa and Keisuke Tanaka. Key dependent message security and receiver selective opening security for identity-based encryption. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10769 of *Lecture Notes in Computer Science*, pages 32–61. Springer, Heidelberg, March 2018. 6

[KW16] Venkata Koppula and Brent Waters. Circular security separations for arbitrary length cycles from LWE. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 681–700. Springer, Heidelberg, August 2016. 2

[KW19] Venkata Koppula and Brent Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume

11693 of *Lecture Notes in Computer Science*, pages 671–700. Springer, Heidelberg, August 2019. 1, 6

[Lev85] Leonid A. Levin. One-way functions and pseudorandom generators. In *17th Annual ACM Symposium on Theory of Computing*, pages 363–365. ACM Press, May 1985. 3

[LQR$^+$19] Alex Lombardi, Willy Quach, Ron D. Rothblum, Daniel Wichs, and David J. Wu. New constructions of reusable designated-verifier NIZKs. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 670–700. Springer, Heidelberg, August 2019. 2

[Rot13] Ron Rothblum. On the circular security of bit-encryption. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 579–598. Springer, Heidelberg, March 2013. 2

[RS92] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, Heidelberg, August 1992. 1

[WZ17] Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under LWE. In Chris Umans, editor, *58th Annual Symposium on Foundations of Computer Science*, pages 600–611. IEEE Computer Society Press, October 2017. 2