How to Build a Trapdoor Function from an Encryption Scheme

Sanjam Garg*, Mohammad Hajiabadi**, Giulio Malavolta***, and Rafail Ostrovsky†

Abstract. In this work we ask the following question: Can we transform any encryption scheme into a trapdoor function (TDF)? Alternatively stated, can we make any encryption scheme randomness recoverable? We propose a generic compiler that takes as input any encryption scheme with pseudorandom ciphertexts and adds a trapdoor to invert the encryption, recovering also the random coins. This *universal TDFier* only assumes in addition the existence of a hinting pseudorandom generator (PRG). Despite the simplicity, our transformation is quite general and we establish a series of new feasibility results:

- The first identity-based TDF [Bellare et al. EUROCRYPT 2012] from the CDH assumption in pairing-free groups (or from factoring), thus matching the state of the art for identity-based encryption schemes. Prior works required pairings or LWE.
- The first collusion-resistant attribute-based TDF (AB-TDF) for all (NC^1, resp.) circuits from LWE (bilinear maps, resp.). Moreover, the first single-key AB-TDF from CDH. To the best of our knowledge, no AB-TDF was known in the literature (not even for a single key) from any assumption. We obtain the same results for predicate encryption.

As an additional contribution, we define and construct a *trapdoor garbling* scheme: A simulation secure garbling scheme with a hidden "trigger" that allows the evaluator to fully recover the randomness used by the garbling algorithm. We show how to construct trapdoor garbling from the DDH or LWE assumption with an interplay of key-dependent message (KDM) and randomness-dependent message (RDM) techniques.

Trapdoor garbling allows us to obtain alternative constructions of (single-key) AB-TDFs with additional desirable properties, such as adaptive security (in the choice of the attribute) and projective keys. We expect trapdoor garbling to be useful in other contexts, e.g. in case where, upon successful execution, the evaluator needs to immediately verify that the garbled circuit was well-formed.

* University of California, Berkeley. supported in part from DARPA/ARL SAFEWARE Award W911NF15C0210, AFOSR Award FA9550-15-1-0274, AFOSR Award FA9550-19-1-0200, AFOSR YIP Award, NSF CNS Award 1936826, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley). The views expressed are those of the author and do not reflect the official policy or position of the funding agencies.

** University of Waterloo.

*** Max Planck Institute for Security and Privacy.

† University of California, Los Angeles. Supported by DARPA SPAWAR contract N66001-15-C-4065.

1 Introduction

Seminal results in 1970's laid the foundations of public-key cryptography by introducing the notion of trapdoor functions [14, 35]. These are families of injective functions, where each function can be computed in the forward direction, but a randomly chosen function is one-way. Moreover, any function in the family can be efficiently inverted using the function's associated trapdoor key.

The historical interest in TDFs stems from this primitive being sufficient for CPA-secure public-key encryption (PKE) schemes. Recent results, however, have substantially changed this perspective, showing TDFs (or extensions thereof) enable many applications which are beyond the reach of traditional public-key encryption techniques [32, 4, 3, 18, 25]. Most notably, a recent result of Hohenberger, Koppula and Waters shows that TDFs generically imply the existence of CCA-secure PKE schemes [25], while whether or not CPA-secure PKE is sufficient is a long standing open problem. Also, recent advances in TDFs have led to many new feasibility results (e.g. rate-1 oblivious transfer) [18].

What makes trapdoor functions a strong primitive is its inversion property: The inversion algorithm recovers the entire input, in contrast to randomized PKE schemes where the decryption algorithm may not necessarily recover the encryption randomness. Such a property is crucially used in all the above applications, and in particular is the central ingredient of the recent result of Hohenberger et al. [25] for enforcing well-formedness of ciphertexts.

We now have constructions of TDFs from a range of specific assumptions, including factoring/QR/DCR [33, 19], LWE/DDH [32], low-noise LPN [27], and CDH [21, 20]. One limitation of these results is that they employ what appear to be ad-hoc techniques to obtain TDFs. Consequently, it is not clear whether and how these techniques will scale to obtain TDFs for more advanced primitives such as Identity-Based Encryption (IBE) or Attribute-Based Encryption (ABE). As we argue below, while TDFs for such advanced primitives will have additional applications, our knowledge of how these advanced TDFs may be realized is quite limited.

TDFs for Advanced Encryption. In this work, we are interested in realizing TDF notions for advanced encryption primitives, such as ABE or IBE. For example, under *Identity-Based TDFs* (IB-TDFs), one *deterministically* evaluates an input $x \in \{0,1\}^n$ as $\mathsf{Eval}(\mathsf{pp}, \mathsf{id}, x)$ to get an image y; an inverter with knowledge of $\mathsf{td}_{\mathsf{id}}$, a user secret key for identity id, may retrieve x as $\mathsf{Invert}(\mathsf{td}_{\mathsf{id}}, y)$. More generally, under *Attribute-Based TDFs* (AB-TDFs), one deterministically evaluates an input x relative to a public parameter pp and attribute α to get an image y; the value of x can be recovered from y by using a trapdoor key td_C for a circuit C where $C(\alpha) = 1$. Moreover, if $C(\alpha) = 0$, then even given td_C it should be computationally infeasible to recover x from y. This notion can be extended to allow many key corruptions, as in the standard ABE setting, or even to hide the attribute if $C(\alpha) = 0$, as in predicate encryption (PE).

Why TDFs for Advanced Encryption? In addition to being an interesting notion on its own, AB-TDFs enable applications that cannot be obtained using

either TDFs or randomized ABE schemes alone. For example, while TDFs allow us to build CCA-secure PKE schemes [25], we do not know whether TDFs are sufficient for realizing stronger primitives, such as Designated-Verifier Non-Interactive Zero-Knowedge (DV-NIZK). On the other hand, the work of [31] shows that DV-NIZK can be realized using any single-key AB-TDFs or, more generally, under what they called single-key weak-function hiding ABE. At a high level, this latter notion requires that using sk_C to decrypt any (possibly malformed) ciphertext c with a public attribute x should reveal nothing beyond whether or not $C(x) = 1$. Any AB-TDF (or equivalently a randomness-recoverable ABE) by design has this property as one can decrypt and re-encrypt to reject all malformed ciphertexts.

Also, the direct usage of IB/AB-TDF can be beneficial: As shown in [5], they directly imply secure constructions of *deterministic* encryption [2, 7] (lifted to the IBE/ABE settings) that allow users to publicly search over encrypted data while maintaining the maximum level of privacy possible. Another application of this class of primitives is the construction of *hedged* IBE/ABE [3, 36] where the encryption algorithm is made resilient to the presence of low-quality randomness.

Assumptions Behind AB-TDFs/IB-TDFs. While single-key ABE can be built from any CPA-secure PKE, we do not have any constructions of (even) single-key AB-TDFs for general circuits. The closest result is the approach of [31], which shows that, assuming LWE, one can build an ABE scheme where the decryption algorithm recovers an "encoded" version of the underlying randomness, which nonetheless is sufficient for checking whether the ciphertext is well-formed (i.e. the LWE noise values sampled during encryption, as opposed to the encryption coins themselves). Even in the simpler IBE setting, the only known constructions of IB-TDFs (for the multi-key setting) are from pairings/LWE [5]. Recent advancements in the IBE landscape showed constructions from CDH/DDH [16, 15, 12], however it is currently not clear whether these assumptions are sufficient for constructing IB-TDFs.

Difficulties in Building AB-TDFs. Let us review the construction of single-key ABE from CPA-secure PKE, as a way to understand the underlying difficulties in realizing AB-TDFs. For circuits of size m, the public parameter $\mathsf{pp} := \{\mathsf{pk}_{i,b}\}_{i\in[m],b\in\{0,1\}}$ contains $2m$ public keys; a secret key for a circuit $C \in \{0,1\}^m$ is the sequence of secret keys $\{\mathsf{sk}_{i,C_i}\}$. To encrypt a message m relative to (pp, α) we garble the universal circuit which has $[\alpha, m]$ hardwired, and which on input C returns m if and only if $C(\alpha) = 1$. The resulting ciphertext consists of the garbled circuit as well as an encryption of the (i, b) label under $\mathsf{pk}_{i,b}$. Making the above scheme randomness recoverable encounters two major difficulties: (1) The random coins used to generate the garbled circuit during encryption should be kept private to ensure circuit privacy; and (2) The random coins used to encrypt the $2m$ labels should also be kept secret, since otherwise the security of the garbled circuit will be lost. Using a TDF (or equivalently a randomness recoverable PKE) to encrypt the labels does not get us too far either, because it will only allow us to recover half of the randomness used to encrypt the labels (and none of the coins used to garble the circuit).

The same obstacles emerge trying to extend the approach of [16, 15] to get IB-TDFs: During encryption we generate a sequence of garbled circuits, and recovering the underlying random coins seems hopeless, as explained above.

1.1 Our Results

We propose a generic approach for building TDFs for advanced encryption functionalities. Our first contribution is the construction of a *universal TDFier*: Given any IBE/ABE/PE with pseudorandom ciphertexts, our compiler returns a secure IB/AB/P-TDF. Our transformation is insensitive to the exact security and functionality of the underlying encryption scheme: For example, if the base ABE is single-key (resp., collusion) secure for a function family F, then so is the resulting AB-TDFs. The only additional building block needed by our compiler is a hinting PRG [29]. More precisely, we prove the following theorem.

Theorem 1 (Informal). *If there exists a {PKE, IBE, ABE, PE} with pseudorandom ciphertext and a hinting PRG, then there exists a {TDF, IB-TDF, AB-TDF,P-TDF}. Moreover, if the hinting PRG is* robust*, the constructed scheme provides deterministic-encryption security.*

To complement our result, we define and construct hinting PRGs that satisfy (k, n)-robustness: The hinting PRG provides pseudorandomness even if the seed is not uniformly distributed, but is a (k, n)-source (i.e., the n-bit seed has at least k-bits min-entropy). By tweaking the hinting-PRG construction of [29], we show the following.

Theorem 2 (Informal). *If the {CDH, LWE} problem is hard, then there exists a (k, n)-robust hinting PRG.*

One of the most surprising features of our compiler is its *simplicity*: Once all components are in place, our transformation adds a very small (conceptual and computational) overhead. Yet, it is quite *general*. It allows us to establish a new series of feasibility results, such as:

- The first IB-TDF from the CDH assumption in pairing-free groups (or under factoring), thus matching the state of the art for IBE schemes. Prior work [5] gave two specialized constructions from pairings and LWE, respectively.
- The first (collusion-resistant) {AB-TDF, P-TDF} for NC^1 circuits from bilinear maps. To the best of our knowledge, this is the first provably secure version of AB-TDF from any assumption.
- The first (collusion-resistant) {AB-TDF, P-TDF} for all circuits from the LWE assumption. We believe that there might already be a way to build (many-key) AB-TDF for circuits from LWE, by using the techniques of [1, 9] in the context of [5]. The resulting construction might be more efficient than our HPRG-based LWE-based one, but ours is generic.

As an added bonus, using our compiler, the public/secret keys of the resulting TDF are identical to that of the underlying encryption scheme. This means that

the trapdoor functionality (i.e. the randomness recoverability) can be added to the encryption *after the fact*, without the need to redistribute keys, by just including some additional public parameters (which can be reused across multiple instances of the scheme).

Trapdoor Projective Garbled Circuits. We also initiate the study of randomness recoverability for garbled circuits. Our result is a new notion of *trapdoor garbled circuits*: Given a garbled circuit $\tilde{\mathsf{P}}$ and garbled labels $\{\ell_{i,b}\}$ produced for a circuit $\mathsf{P} : \{0,1\}^m \to \{0,1\}$ using randomness r, we require the following properties. (1) Randomness recoverability: For any x such that $\mathsf{P}(x) = 1$, given $(\tilde{\mathsf{P}}, \{\ell_{i,x_i}\})$ we can recover r; and (2) Privacy: We have simulation security against any input x such that $\mathsf{P}(x) = 0$. We note that a randomness recoverable single-key ABE (constructed from Theorem 1 above) allows one to build a trapdoor garbled circuit, but it will not be projective (i.e., one label for each input wire value). The projective property of garbled circuits is crucially used in many applications, and hence is a desirable feature to have. One of our main contributions is a construction of trapdoor garbling under standard assumptions.

Theorem 3 (Informal). *If the {DDH, LWE} problem is hard, then there exists a trapdoor (projective) garbling scheme for all circuits.*

Our scheme builds on an interplay of key-dependent message (KDM) and randomness-dependent message (RDM) techniques, which may be of independent interest. As an immediate application, we obtain a single-key AB-TDF that is simultaneously (1) adaptively secure (in the choice of the identity/attribute), (2) perfectly correct, and (3) has projective keys (i.e. one label for each input wire value).[1] We believe trapdoor garbling may find further applications in the future.

2 Technical Overview

In this section we give an overview of our techniques for building TDFs from advanced encryption schemes.

2.1 A Universal TDFier

Our construction of universal TDFier is quite simple and the best way to present it is to just describe the scheme. It builds on the mirroring technique of [20]. Before delving into the technical details, we recall the notion of a hinting PRG and we briefly define the security notion of (k, n)-robustness that is at the heart of our transformation.

Robust Hinting PRGs. The notion of hinting PRG [29] was introduced in the context of upgrading CPA-secure scheme into CCA-secure ones.[2] A hinting

[1] Our generic conversion starting from any single-key ABE does not preserve property (1) and (3).

[2] Although there is some resemblance with our approach, we note that the compiled scheme of [29] is *not* randomness recoverable unless one starts with a randomness recoverable encryption scheme, which is tautological.

PRG, is a function that takes as input an n-bit seed and returns n-many ℓ-bit strings, for some polynomial $\ell = \ell(\lambda)$. A hinting PRG is required to satisfy an enhanced notion of pseudorandomness. In the security game, we define a 2-by-n matrix where the rows corresponding to the bit representation of the seed are populated with the outputs of the hinting PRG, whereas the other entries are sampled uniformly ℓ-bit strings. The requirement is that it is computationally hard to distinguish such 2-by-n matrix from a uniform one. Note that such a notion does not follow from the standard definition of pseudorandomness: The pseudorandom entries of the 2-by-n matrix give a "hint" about the input seed. Nevertheless, it was shown that such a notion can be achieved under standard assumptions, such as CDH or LWE [29].

In this work we are interested in a stronger notion of security that we call (k, n) robustness. Loosely speaking, we require that the above guarantee is retained even if the n-bit seed is not sampled uniformly, but rather from a distribution with k-bits of min entropy. We observe that the schemes in [29], which are in turn based on the constructions of chameleon encryption from [13, 16, 17, 12, 21], can be shown to satisfy (k, n)-robustness with some tweaking of the parameters and of the security analysis. Henceforth, we simply assume we are given a (k, n)-robust hinting PRG.

Universal TDFier. We are now ready to show how to add randomness recoverability to any encryption scheme (i.e. construction a TDF). We only require such an encryption to satisfy a mild structural property, which is usually referred to as *pseudorandom ciphertext*: Ciphertexts $c \in \mathsf{Enc}(\mathsf{pk}, \cdot)$ must be computationally indistinguishable from uniformly sampled bitstrings $\{0,1\}^{|c|}$. Our actual compiler will be slightly more general, allowing us to capture all ciphertext domains that form a (not necessarily Abelian) group, with efficiently samplable elements. This will allow us to capture a large class of encryption schemes and will significantly expand the scope of our compiler. However, for the sake of this overview we assume ciphertexts are indistinguishable from uniformly random strings.

We describe our universal TDFier for the simple case of PKE, which already contains the main ideas of our approach. We refer the reader to the main body for the generalization to IBE/ABE/PE. The key generation algorithm consists of sampling a key pair $(\mathsf{sk}, \mathsf{pk})$ of the input PKE, along with the public parameters $\mathsf{pp}_{\mathsf{HPRG}}$ of a (k, n)-robust hinting PRG and n random strings $r_i \xleftarrow{\$} \{0,1\}^{|c|}$. The index key of the TDF consists of

$$(\mathsf{pk}, \mathsf{pp}_{\mathsf{HPRG}}, r_1, \ldots, r_n)$$

and the trapdoor is set to sk. On input some string $x \in \{0,1\}^n$, we define the evaluation algorithm of the TDF as follows. For all $i \in [n]$:

- If $x_i = 0$: Compute $y_i = \mathsf{Enc}(\mathsf{pk}, \$; z_i)$, where $z_i = \mathsf{HPRG.Eval}(\mathsf{pp}_{\mathsf{HPRG}}, x, i)$.
- If $x_i = 1$: Compute $y_i = \mathsf{Enc}(\mathsf{pk}, \$; z_i) \oplus r_i$, where $z_i = \mathsf{HPRG.Eval}(\mathsf{pp}_{\mathsf{HPRG}}, x, i)$.

Return the image $y = (y_1, \ldots, y_n)$. Here \$ denotes some distinguished string, which is instrumental to ensure correctness of the inversion. Given the trapdoor

sk, one can recover the input x from an image $y = (y_1, \ldots, y_n)$ bit by bit, setting $x_i = 0$ if $\mathsf{Dec}(\mathsf{sk}, y_i) = \$$ and $x_i = 1$ otherwise. Setting $\$$ to be a large enough string, we can show that the scheme achieves perfect correctness with all but negligible probability, over the random choice of the index key.

Besides adding small overhead to the runtime of the encryption scheme (i.e. n evaluations of a hinting PRG and n calls to the encryption algorithm), the public/secret keys of the compiled scheme are identical to that of the underlying PKE, except for some additional public parameters $(\mathsf{pp}_{\mathsf{HPRG}}, r_1, \ldots, r_n)$. This means that we can take any encryption scheme (with pseudorandom ciphertexts) and add randomness recoverability almost for free.

Security Proof (Sketch). We provide a high-level idea of the proof strategy, to motivate the security requirements for the underlying building blocks. To prove CPA-security, we modify the distribution of the challenge image through a series of hybrids, which we summarize below.

- Hybrid 0: This is the original distribution.
- Hybrid 1: In this hybrid we compute the public parameters after the challenge ciphertext. More precisely, for all $i \in [n]$:
  - If $x_i = 0$: Compute $y_i = \mathsf{Enc}(\mathsf{pk}, \$; z_i)$, where $z_i = \mathsf{HPRG.Eval}(\mathsf{pp}_{\mathsf{HPRG}}, x, i)$. The set $r_i = y_i \oplus s_i$, where $s_i \xleftarrow{\$} \{0,1\}^{|c|}$
  - If $x_i = 1$: Compute $y_i = s_i \xleftarrow{\$} \{0,1\}^{|c|}$ and set $r_i = y_i \oplus \mathsf{Enc}(\mathsf{pk}, \$; z_i)$, where $z_i = \mathsf{HPRG.Eval}(\mathsf{pp}_{\mathsf{HPRG}}, x, i)$.

  This step is reminiscent of the mirroring technique from [20] and will allow us to later equivocate the challenge image. Note that so far the distribution did not change.
- Hybrid 2: In this hybrid we change the s_i (as defined above) to be encryptions of $\$$ with fresh random coins. Indistinguishability follows from the pseudorandomness of the ciphertexts of the encryption scheme. The effect of this change is to remove the "signal" of x in the challenge image: Regardless of the value of x_i, we always compute y_i as an encryption of $\$$ and r_i as $\mathsf{Enc}(\mathsf{pk}, \$) \oplus \mathsf{Enc}(\mathsf{pk}, \$)$. However we are not yet done: The usage of pseudorandom/truly random coins still contains some lingering information about x.
- Hybrid 3: In this hybrid we compute all ciphertexts for the challenge image with truly random coins. It is tempting to conclude that the indistinguishability follows from the pseudorandomness of the hinting PRG, however note that x is not necessarily uniformly sampled.[3] Our final weapon to deploy is the notion of (k, n)-robustness of the hinting PRG: Provided that x has enough min-entropy, this step goes trough.
- Hybrid 4: In the last step we change all ciphetexts to encrypt 0 (padded to the appropriate length). Since they are all computed using truly random coins, we can now invoke the CPA-security of the encryption scheme.

[3] One could define the security of the TDF to hold only for uniformly sampled inputs (i.e. one-wayness) however this precludes many interesting applications, such as deterministic and searchable encryption.

The proof is concluded by observing that the image in the last hybrid has no pre-image.

2.2 Trapdoor Garbled Circuits and Single-Key ABE

We will now show an alternative way of building single-key AB-TDFs. We will start with a single-key ABE built from a CPA-secure PKE and garbled circuits and develop new techniques that will allow us to make this scheme randomness recoverable. This new approach has several advantages when compared with our generic transformation:

- The resulting AB-TDF is adaptively secure in the choice of the attribute string α.
- The resulting AB-TDF has projective keys, i.e. one labels per input wire.

Along the way, we define and construct a notion we call *trapdoor garbling*, which may be of independent interest.

An ABE scheme ABE is defined as follows. The public parameter $\mathsf{pp} := \{\mathsf{pk}_{i,b}\}$ consist of $2n$ public keys; the secret key for a circuit C is $\mathsf{sk}_C := \{\mathsf{sk}_{i,C_i}\}$. To encrypt a message m to an attribute α, we garble the universal circuit $\mathsf{P}[\alpha, m]$ (where $\mathsf{P}[\alpha, m](C) = m$ iff $C(\alpha) = 1$) to get $(\tilde{\mathsf{P}}, \{\mathsf{lb}_{i,b}\})$; we then output $\mathsf{ct} := (\tilde{\mathsf{P}}, \{\mathsf{PKE.Enc}(\mathsf{pk}_{i,b}, \mathsf{lb}_{i,b})\})$. Let ρ be the randomness used to garble $C[\alpha, m]$ and let $\{r_{i,b}\}$ be the randomness used to encrypt the labels $\{\mathsf{lb}_{i,b}\}$.

How to Make ABE Randomness Recoverable? Recall the two sources of randomness ρ, $\{r_{i,b}\}$ mentioned above. At first, one might think that recovering ρ would be too much to ask for, since otherwise the whole security of the garbled circuit is lost. We, however, notice that there is some wiggle room here: Only *legitimate* inverters — namely one who has sk_C where $C(\alpha) = 1$ — need to be able to recover ρ, while security should hold against *illegitimate* inverters. This brings us to the notion of *trapdoor garbling*.

Trapdoor Garbling. We explain the idea of trapdoor garbling in the above context. Letting $(\tilde{\mathsf{P}}, \{\mathsf{lb}_{i,b}\})$ be as above, trapdoor garbling requires: (1) Randomness Recoverability: for any C such that $C(\alpha) = 1$, given $(\tilde{\mathsf{P}}, \{\mathsf{lb}_{i,C_i}\})$ one may efficiently recover ρ; we call this *trapdoor mode*; and (2) Security: for any C such that $C(\alpha) = 0$ (which we call *security mode*), the pair $(\tilde{\mathsf{P}}, \{\mathsf{lb}_{i,C_i}\})$ can be simulated in the standard sense (and in particular without knowing m or α).

Realizing Trapdoor Garbling. Let us see the challenges involved in adding a trapdoor mode to Yao's garbling scheme. For Yao's scheme, the garbling randomness ρ may be split into (A) ρ_k: the coins used to generate all the underlying wire keys and (B) ρ_c: the coins used to generate the underlying ciphertexts for the garble tables. Yao's scheme guarantees coin recovery for neither case. Let Key be the set of all keys produced during the garbling algorithm (i.e., two keys per wire). Our first observation is that if the underlying secret-key encryption scheme $\mathsf{SKE} := (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ is randomness recoverable and that if a secret key is just the coins of G, then recovering *all* the keys in Key will enable recovering both ρ_k and ρ_c. We can recover ρ_k because a key is just the coins of G;

and we can recover ρ_c because we can decrypt every ciphertext to recover the underlying coins. Now to recover Key in trapdoor mode, letting $k_{\mathsf{out},1}$ be the output-wire key for value 1, we simply add an encryption $\mathsf{SKE.Enc}(k_{\mathsf{out},1}, k)$ for any $k \in \mathsf{Key}$ to $\tilde{\mathsf{P}}$.[4] We will have randomness recoverability, but we have introduced key-dependent-message (KDM) circularity involving multiple keys (since $k_{\mathsf{out},1}$ encrypts all the keys in Key, and is in turn encrypted under those keys via a chain of "hops"). While we have multi-key KDM-secure SKE schemes, we have to make sure both Conditions (A) and (B) above hold. Fortunately, the DDH-based SKE schemes of Boneh et al. ($\mathsf{SKEBHHO}$) [10] and the LWE-based Dual-Regev's SKE scheme [34, 22] provide both Conditions (A) and (B). For concreteness, under $\mathsf{SKEBHHO}$, a key is chosen as $s \xleftarrow{\$} \{0,1\}^n$; to encrypt $\mathsf{Enc}(s, g)$: we return $(g_1, \ldots, g_n, s \cdot \mathbf{g})$, where $\mathbf{g} := (g_1, \ldots, g_n) \xleftarrow{\$} \mathbb{G}^n$ and $s \cdot \mathbf{g} = \prod g_i^{s_i}$.[5] The proof of security is exactly like Yao's scheme [37, 30], breaking the circularity using the underlying KDM-secure scheme.

ABE Made Randomness Recoverable? Unfortunately, we are not done yet, because in the ABE scheme we need to recover both ρ (the garbled circuit randomness) and $\{r_{i,b}\}$, the randomness used to encrypt the labels $\{\mathsf{lb}_{i,b}\}$ under $\{\mathsf{pk}_{i,b}\}$. Trapdoor garbling allows us to recover ρ, but we are left with recovering $\{r_{i,b}\}$. Even if the underlying PKE scheme is randomness recoverable, we can only recover half of $\{r_{i,b}\}$, those corresponding to the bits of C. Moreover, we cannot make $r_{i,0}$ and $r_{i,1}$ related (so to have either one reveal the other one) because the ABE security will be lost. So, it seems we are stuck here. To get around this, we augment the garbled circuit $\tilde{\mathsf{P}}$ even further! We now add to $\tilde{\mathsf{P}}$ an encryption of $\mathsf{SKEBHHO.Enc}(k_{\mathsf{out},1}, r_{i,b})$ for all i and b, where recall that $k_{\mathsf{out},1}$ is the output-wire key for bit 1. Doing so will indeed make the underlying ABE scheme randomness recoverable, but we have now introduced a much more complicated circularity, involving both KDM and RDM (randomness-dependent messaging) at the same time. In particular, (i) $\mathsf{pk}_{i,b}$ encrypts $\mathsf{lb}_{i,b}$ using coins $r_{i,b}$ and (ii) $r_{i,b}$ is encrypted under $k_{\mathsf{out},1}$; and (iii) $k_{\mathsf{out},1}$ is encrypted under $\mathsf{lb}_{i,b}$ (via a sequence of intermediate encryptions). We have RDM dependence because of (ii) and similarly we have KDM dependence. We now introduce a technique that will allow us to handle the above circularity, using a careful choice of encryption schemes. To make things more concrete, let us focus on a special case of the above circularity which nonetheless captures all the difficulties: (A) $\mathsf{pk}_{i,b}$ encrypts $k_{\mathsf{out},1}$ using coins $r_{i,b}$ and (B) $k_{\mathsf{out},1}$ encrypts $r_{i,b}$ using fresh coins (under the SKE scheme).

[4] In Yao's scheme [37, 30], the key $k_{\mathsf{out},1}$ is put in the clear in $\tilde{\mathsf{P}}$ with the corresponding bit 1 for it; we do not put $k_{\mathsf{out},1}$ in the clear in $\tilde{\mathsf{P}}$, but rather we encrypt 1 under $k_{\mathsf{out},1}$ to assert the underlying recovered key corresponds to bit 1; similarly, we encrypt 0 under the output-wire key $k_{\mathsf{out},0}$ for bit 0.

[5] One might complain that the scheme is not randomness recoverable in a strict sense, in that the coins used to sample the group elements are not recovered. We note, however, that in our ABE application, these group elements $\mathbf{g}$ may be chosen during key-generation time and put in pp. We ignore these issues for simplicity.

Handling RDM+KDM. Let $\mathsf{SKE} := (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ be an SKE scheme and $\mathsf{PKE} := (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme. We need security in the presence of

$$\mathsf{pk}, \mathsf{PKE.Enc}(\mathsf{pk}, s; r), \mathsf{SKE.Enc}(s, r), \tag{1}$$

where all the variables are chosen at random and that $\mathsf{SKE.Enc}$ uses fresh coins to encrypt. By having security, we mean semantic security against both pk and s. We give a technique that allows us to reduce RDM+KDM in the sense of Equation 1 to KDM alone. Focusing on DDH, SKE will be $\mathsf{SKEBHHO}$ (see above) and PKE will be the dual version of $\mathsf{PKE}_{\mathrm{bhho}}$, where the roles of randomness and secret keys are swapped.

Let $|s| = n$. Let $\mathsf{PKE}_{\mathrm{bhho}}$ be the n-bit version of BHHO (defined below). We will define a dual version of n-bit BHHO, which we call $\mathsf{PKE}_{\mathrm{dbhho}}$, satisfying:

1. An encryption randomness under $\mathsf{PKE}_{\mathrm{dbhho}}$ is a secret key under $\mathsf{PKE}_{\mathrm{bhho}}$. Also, a secret key under $\mathsf{PKE}_{\mathrm{dbhho}}$ is an encryption randomness under $\mathsf{PKE}_{\mathrm{bhho}}$.
2. Looking at the PKE part of Equation 1, we require

$$\mathsf{pk}_{\mathrm{dbhho}}, \mathsf{PKE}_{\mathrm{dbhho}}.\mathsf{Enc}(\mathsf{pk}_{\mathrm{dbhho}}, s; \mathsf{r}_{\mathrm{dbhho}}) \equiv \mathsf{pk}_{\mathrm{bhho}}, \mathsf{PKE}_{\mathrm{bhho}}.\mathsf{Enc}(\mathsf{pk}_{\mathrm{bhho}}, s; \mathsf{r}_{\mathrm{bhho}}), \tag{2}$$

where $(\mathsf{pk}_{\mathrm{dbhho}}, \mathsf{sk}_{\mathrm{dbhho}})$ is a key pair under $\mathsf{PKE}_{\mathrm{dbhho}}$, $\mathsf{sk}_{\mathrm{bhho}} := \mathsf{r}_{\mathrm{dbhho}}$, $\mathsf{pk}_{\mathrm{bhho}}$ is the corresponding public key for $\mathsf{sk}_{\mathrm{bhho}}$ under $\mathsf{PKE}_{\mathrm{bhho}}$ and $\mathsf{r}_{\mathrm{bhho}} := \mathsf{sk}_{\mathrm{dbhho}}$.

Instantiating Equation 1 with $\mathsf{PKE}_{\mathrm{dbhho}}$, using Equation 2, we may rewrite Equation 1 as

$$\mathsf{pk}_{\mathrm{bhho}}, \mathsf{PKE}_{\mathrm{bhho}}.\mathsf{Enc}(\mathsf{pk}_{\mathrm{bhho}}, s; \mathsf{r}_{\mathrm{bhho}}), \mathsf{SKEBHHO.Enc}(s, \mathsf{sk}_{\mathrm{bhho}}), \tag{3}$$

where everything is chosen randomly. Since the randomness of $\mathsf{PKE}_{\mathrm{bhho}}.\mathsf{Enc}$ (i.e., $\mathsf{r}_{\mathrm{bhho}}$) never appears as a plaintext, Equation 3 is secure by [10].[6]

We quickly review k-bit BHHO [10], where the secret key size is n. (In our ABE instantiation, k will be n, because we need to encrypt a secret key of the SKE scheme.) We have a public parameter $\mathbf{g} := (g_1, \ldots, g_n)$. The secret key of $\mathsf{PKE}_{\mathrm{bhho}}$ is $\mathsf{sk} \in \{0,1\}^n$ and the public key is $(\mathbf{g}, g_{\mathsf{pub}})$, where $g_{\mathsf{pub}} := \mathsf{sk} \cdot \mathbf{g}$. To encrypt k group elements $(g'_1, \ldots, g'_k)$, sample k exponents $\mathbf{r} := (f_1, \ldots, f_k)$ and set $\mathsf{ct} := (\mathbf{g}^{f_1}, g_{\mathsf{pub}}^{f_1} \cdot g'_1, \ldots, \mathbf{g}^{f_k}, g_{\mathsf{pub}}^{f_k} \cdot g'_k)$.

Dual BHHO. We define k-bit $\mathsf{PKE}_{\mathrm{dbhho}}$ as follows: the secret key is k random exponents $(f_1, \ldots, f_k)$ and the public key is $(\mathbf{g}, \mathbf{g}'_1, \ldots, \mathbf{g}'_k) := (\mathbf{g}, \mathbf{g}^{f_1}, \ldots, \mathbf{g}^{f_k})$, where $\mathbf{g}$ is as above. To encrypt k-group elements $(g'_1, \ldots, g'_k)$, sample $s \xleftarrow{\$} \{0,1\}^n$ and return $(s \cdot \mathbf{g}, (s \cdot \mathbf{g}'_1) \times g'_1, \ldots, (s \cdot \mathbf{g}'_k) \times g'_k)$.

Now notice that an encryption randomness under k-bit $\mathsf{PKE}_{\mathrm{dbhho}}$ is an n-bit string, a secret key for k-bit $\mathsf{PKE}_{\mathrm{bhho}}$. Also, a secret key for k-bit $\mathsf{PKE}_{\mathrm{dbhho}}$ is

[6] The result of [10] concerns a single PKE scheme; in our setting $\mathsf{SKEBHHO}$ is just the secret-key version of BHHO, and by choosing the public parameter to be the same across the two schemes, we will have cross KDM security.

a tuple of k exponents, an encryption randomness for k-bit $\mathsf{PKE}_{\mathrm{bhho}}$. Finally, Equation 2 may be verified by inspection. (See Lemma 5.)

For LWE we can do a similar trick, by plugging in Regev's PKE scheme [34] and Dual-Regev's secret-key scheme [22] in Equation 1. While dual PKE/SKE Regev is known, we are the first to introduce Dual BHHO. We believe our RDM/KDM switching technique may find other applications in the future.

Back to the AB-TDF scheme, Equation 1 may now be used to reduce RDM+KDM (created by encrypting ρ and $\{r_{i,b}\}$ under $k_{\mathsf{out},1}$) to a KDM-only setting. At this point, we can rely on the underlying KDM-secure schemes to argue security for the garbled circuit.

How is Adaptive Security Obtained? We now have a randomness-recoverable single-key ABE scheme. Unlike the construction in Section 2.1 which only provides selective security, we obtain adaptive security in the sense that the attribute α may be chosen based on the ABE public parameter. We get this exactly because of the same reason that single-key ABE constructed from CPA-secure PKE provides adaptive security in the choice of α. The difference in our setting is that (after the RDM+KDM to KDM reduction) we have KDM-dependency, but notice that the input wires for the corrupted circuit C are chosen *non-adaptively*; this is why we can use the KDM results as is. In other words, the adaptive choice of α, only specifies the circuit $\mathsf{P}[\alpha]$, but the input C to this circuit is chosen non-adaptively, and not after seeing $\tilde{\mathsf{P}}$.

2.3 Related Works

The work of Kitagawa et al. [28] shows a generic approach for constructing TDFs, starting from randomness-recoverable KDM-secure symmetric encryption and PKE with pseudorandom ciphertexts. We highlight the main conceptual and technical differences in the following.

- **Robustness:** The TDF from [28] is not robust, i.e. it does not offer deterministic-encryption security. This means that none of our results on deterministic IBE/ABE follows from their work. This is not just an artifact of the analysis, in fact there is a concrete attack[7] if one allows non-uniform (but high-entropy) inputs: In their scheme, a domain point is of the form $(s, r_1, \ldots, r_n)$. Fixing a distribution where the first $n/2$ bits of s and the variables $(r_1, \ldots, r_{n/2})$ are fixed to 0, we obtain an image that contains

  $$\mathsf{Enc}(\mathsf{pk}, 0; 0), \ldots, \mathsf{Enc}(\mathsf{pk}, 0; 0), \mathsf{Enc}(\mathsf{pk}, s_{n/2+1}; r_{n/2+1}), \ldots, \mathsf{Enc}(\mathsf{pk}, s_n; r_n)$$

  which is easily distinguishable from uniform. The high-level issue (that our approach overcomes using robust hinting PRGs) is the *locality* of the output bits of the TDF.

[7] We remark that this attack does not invalidate any claim made in [28], but rather exemplifies the separation between their approach and ours.

- **Generality:** The work of [28] does not elaborate on more advanced primitives than TDFs. Although it appears to be likely that one could generalize their techniques to construct IB/AB-TDF, they would suffer from the above mentioned drawbacks. Furthermore, we view the connection of hinting PRGs with IB/AB-TDF as an important conceptual contribution of our work.
- **Assumptions:** Our work requires hinting PRGs, whereas [28] assumes randomness-recoverable KDM-secure symmetric encryption. These assumptions are not known to be equivalent: From hinting PRGs, one can build KDM-secure encryption, but it is not randomness recoverable.

Finally, we mention that [31] showed[8] a construction of trapdoor garbling (although with a different syntax) for NC^1 circuits, assuming CDH or LWE. Their approach does not appear to be extendable to all circuits, due to the reliance on information-theoretic secret-sharing.

3 Preliminaries

Notation. We use λ for the security parameter. We use $\stackrel{c}{\equiv}$ to denote computational indistinguishability and use $\equiv$ to denote two distributions are identical. For a distribution $\mathcal{S}$ we use $x \xleftarrow{\$} \mathcal{S}$ to mean x is sampled according to $\mathcal{S}$ and use $y \in \mathcal{S}$ to mean $y \in \sup(\mathcal{S})$, where sup denotes the support of a distribution. For a set S we overload the notation to use $x \xleftarrow{\$} \mathsf{S}$ to indicate that x is chosen uniformly at random from S. If $\mathsf{A}(x_1, \ldots, x_n)$ is a randomized algorithm, then $\mathsf{A}(a_1, \ldots, a_n)$, for deterministic inputs $a_1, \ldots, a_n$, denotes the random variable obtained by sampling random coins r uniformly at random and returning $\mathsf{A}(a_1, \ldots, a_n; r)$. We use $[n] := \{1, \ldots, n\}$ and $[i, i+s] := \{i, i+1, \ldots, i+s\}$. The min-entropy of a distribution $\mathcal{S}$ is defined as $\mathsf{H}_\infty(\mathcal{S}) \stackrel{\triangle}{=} -\log(\max_{\mathsf{x}} \Pr[\mathcal{S} = \mathsf{x}])$. We call a distribution $\mathcal{S}$ a (k, n)-source if $\mathsf{H}_\infty(\mathcal{S}) \geq k$ and $\sup(\mathcal{S}) \subseteq \{0,1\}^n$. We recall the standard notion of statistical distance. Unless otherwise stated, we assume the length of a randomness value to a function is λ.

Definition 1 (Statistical Distance). *Let $\mathcal{X}$ and $\mathcal{Y}$ be two random variables over a finite set U. The statistical distance between $\mathcal{X}$ and $\mathcal{Y}$ is defined as*

$$\mathbb{SD}[\mathcal{X}, \mathcal{Y}] = \frac{1}{2} \sum_{u \in \mathsf{U}} \left| \Pr[\mathcal{X} = u] - \Pr[\mathcal{Y} = u] \right|.$$

3.1 Standard Lemmas

We recall the Markov inequality.

Lemma 1 (Markov Inequality). *Let $\mathcal{X}$ be a non-negative random variable. Then, for all $\varepsilon > 0$:*

$$\Pr[\mathcal{X} \geq \varepsilon] \leq \frac{\mathsf{E}[\mathcal{X}]}{\varepsilon}$$

where $\mathsf{E}[\mathcal{X}]$ denotes the expected value of $\mathcal{X}$.

[8] The scheme appears in Appendix D in an older version of the paper.

We recall the definition of universal hash and the leftover hash lemma [26].

Definition 2 (Universal Hash Functions). *An ensemble of functions* $\mathcal{H} : \mathsf{X} \to \mathsf{Y}$ *is called universal, if it holds for all* $x \neq x' \in \mathsf{X}$ *that*

$$\Pr_{h}[h(x) = h(x')] \leq 1/|\mathsf{Y}|$$

where $h \xleftarrow{\$} \mathcal{H}$.

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

$$\mathbb{SD}[(h(\mathcal{S}, \mathcal{X}), \mathcal{S}), (\mathcal{U}, \mathcal{S})] \leq \varepsilon$$

where $\mathcal{S}$ *is uniform over* S *and* $\mathcal{U}$ *is uniform over* Y.

3.2 Standard TDFs

We recall the notion of trapdoor functions (TDFs).

Definition 3 (Trapdoor Functions). *Let* $n = n(\lambda)$ *be a polynomial. A family of trapdoor functions* $(\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Invert})$ *with domain* $\{0,1\}^n$ *consists of the following algorithms.*

- $\mathsf{KeyGen}(1^\lambda)$*: On input the security parameter, the key generation algorithm returns the index key* ik *and the trapdoor* td.
- $\mathsf{Eval}(\mathsf{ik}, x)$*: On input the index key* ik *and an input string* $x \in \{0,1\}^n$, *the evaluation algorithm returns an image* y.
- $\mathsf{Invert}(\mathsf{td}, y)$*: On input a trapdoor* td *and an image* y, *the inversion algorithm returns a pre-image* x.

We require the following properties.

- *(Correctness) There exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$ *it holds that*

$$\Pr_{(\mathsf{ik},\mathsf{td})}[\exists x \in \{0,1\}^n \text{ s.t. } \mathsf{Invert}(\mathsf{td}, \mathsf{Eval}(\mathsf{ik}, x)) \neq x] = \mathsf{negl}(\lambda)$$

  where the probability is taken over $(\mathsf{ik}, \mathsf{td}) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$.
- *(One-Wayness) There exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$ *and all PPT adversaries* $\mathcal{A}$ *it holds that*

$$\Pr[\mathcal{A}(\mathsf{ik}, \mathsf{Eval}(\mathsf{ik}, x)) = x] = \mathsf{negl}(\lambda)$$

  where $(\mathsf{ik}, \mathsf{td}) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$ *and* $x \xleftarrow{\$} \{0,1\}^n$.

3.3 Predicate Trapdoor Functions

In the following we define a generalized notion of TDFs, that we call predicate TDFs. A predicate TDF allows one to evaluate a function with respect to an attribute α and issue trapdoors for circuits C. The trapdoor td_C allows one to invert the function if and only if $C(\alpha) = 1$ and otherwise the input (as well as the attribute) is hard to recover. This notion generalizes identity-based TDFs [5] in the same way as functional encryption [11] generalizes identity-based [8] and attribute-based encryption [24]. We give a formal definition below.

Definition 4 (Predicate TDFs). *Let $n = n(\lambda)$ be a polynomial. A family of predicate TDFs $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Invert})$ with domain $\{0,1\}^n$ consists of the following algorithms.*

- $\mathsf{Setup}(1^\lambda)$*: On input the security parameter, the setup algorithm returns a master secret key* msk *and some public parameters* pp*.*
- $\mathsf{KeyGen}(\mathsf{msk}, C)$*: On input the master secret key* msk *and a circuit C, the key generation algorithm returns a trapdoor* td_C*.*
- $\mathsf{Eval}(\mathsf{pp}, \alpha, x)$*: On input the public parameters* pp*, an attribute α, and an input string $x \in \{0,1\}^n$, the evaluation algorithm returns an image y.*
- $\mathsf{Invert}(\mathsf{td}_C, y)$*: On input a trapdoor* td_C *for a circuit C and an image y, the inversion algorithm returns a pre-image x.*

We require the following notion of correctness.

- *(Correctness) There exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$ it holds that*

$$\Pr_{(\mathsf{pp},\mathsf{msk})}\left[\exists(x,\alpha,C) \text{ s.t. } \mathsf{Invert}(\mathsf{KeyGen}(\mathsf{msk},C),\mathsf{Eval}(\mathsf{pp},\alpha,x)) \wedge C(\alpha)=1\right] = \mathsf{negl}(\lambda)$$

where the probability is taken over $(\mathsf{pp},\mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*.*

Deterministic Security. We now define a strong notion of security for predicate TDFs, i.e. we require that evaluating the TDF over two inputs with enough min-entropy yields two computationally indistinguishable distributions. This is the natural generalization of the standard notion of security for deterministic encryption [7] and thus we refer to it as *deterministic security*. Note that the following definitions assume without loss of generality that the key generation algorithm is deterministic. This can always be enforced by drawing the random coins from a PRF applied to the input circuit C.

Definition 5 (Deterministic CPA Security). *A predicate TDF is (k,n)-CPA secure if there exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$, all PPT adversaries $\mathcal{A}$, any two (k,n)-sources $\mathcal{S}_0$ and $\mathcal{S}_1$, all pairs of attributes (α_0,α_1) it holds that*

$$\Pr\left[\mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{pp},\alpha_0,\alpha_1,\mathsf{Eval}(\mathsf{pp},\alpha_b,x_b)) = b\right] - 1/2 = \mathsf{negl}(\lambda)$$

where $(\mathsf{pp},\mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, $b \xleftarrow{\$} \{0,1\}$, $x_b \xleftarrow{\$} \mathcal{S}_b$, and the adversary never queries the* KeyGen *oracle on some C such that $C(\alpha_0) = 1$ or $C(\alpha_1) = 1$.*

To draw an analogy to the standard public-key settings (i.e. encryption without randomness recovery) the above definition corresponds to the notion of one-sided security for predicate encryption (where the attribute is not hidden if $C(\alpha) = 1$). However, in our settings this seems to be the best-possible notion to achieve: Since the secret keys for accepting predicates are required to recover all random coins of the evaluation function, it is impossible to fully hide the attribute α if $C(\alpha) = 1$. This is because one can always try to recompute the ciphertext with a candidate attribute and see whether the result matches.

Selective vs Adaptive Security. We note that the definition as stated above captures the *selective* variant of security, where the challenge attributes are fixed ahead of times and prior to the adversary seeing the public parameters of the scheme. The stronger (and perhaps more natural) notion of *adaptive* security allows the adversary to choose the challenge attributes (α_0, α_1) depending on the public parameters of the scheme and possibly the answers of some queries to the KeyGen oracle. The formal definition is the same as Definition 5 modified in the natural way. We remark that any selectively secure scheme can be shown to be also adaptively secure although with an exponential decrease in the success probability of the reduction (via complexity leveraging).

4 Robust Hinting PRGs

A hinting pseudorandom generator (PRG) is a notion introduced in [29] and has since found several applications (e.g. [31, 28]). Roughly speaking, it stretches the input n-bit seed into a $n \cdot \ell$-bit string. In the security game, the distinguisher is given a 2-by-n matrix where the entries corresponding to the seed are taken from the output of the hinting PRG and the others are uniformly sampled. The distinguisher has to tell this distribution apart from a uniformly random 2-by-n matrix. In this work we are interested in a stronger notion of hinting PRG where the seed is not required to be uniformly sampled, instead we only impose that it has high-enough min-entropy. We call this notion *robust* hinting PRG and we provide formal definitions in the following. We recall the syntax of hinting PRG [29].

Definition 6 (Hinting PRGs). *Let $n = n(\lambda)$ and $\ell = \ell(\lambda)$ be two polynomials. A family of hinting PRGs consists of the following algorithms.*

- $\mathsf{Setup}(1^\lambda)$*: On input the security parameter 1^λ, the setup algorithm returns the public parameters* pp*.*
- $\mathsf{Eval}(\mathsf{pp}, x, i)$*: On input the public parameters* pp*, an input string $x \in \{0,1\}^n$, and an index i, the evaluation algorithm returns an image $y \in \{0,1\}^\ell$.*

The security that we require is essentially identical to that of [29], except that we only require the seed to have high min-entropy, as opposed to be uniformly sampled. We name this notion (k, n)-robustness and we present a formal definition below.

Definition 7 (Robustness). *A hinting PRG* $(\mathsf{Setup}, \mathsf{Eval})$ *is* (k, n)*-robust if there exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$*, all PPT adversaries* $\mathcal{A}$*, all* (k, n)*-sources* $\mathcal{S}$ *it holds that*

$$\Pr\left[\mathcal{A}\left(\mathsf{pp}, y_0^b, \begin{pmatrix} y_{1,0}^b, \ldots, y_{n,0}^b \\ y_{1,1}^b, \ldots, y_{n,1}^b \end{pmatrix}\right) = b\right] - 1/2 = \mathsf{negl}(\lambda)$$

where

- $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*,* $x \xleftarrow{\$} \mathcal{S}$*, and* $b \xleftarrow{\$} \{0,1\}$*.*
- $y_0^0 \xleftarrow{\$} \{0,1\}^\ell$ *and* $y_0^1 \leftarrow \mathsf{Eval}(\mathsf{pp}, x, 0)$*.*
- *For all* $i \in [n]$: $\left(y_{i,0}^0, y_{i,1}^0\right) \xleftarrow{\$} \{0,1\}^{2\ell}$*,* $y_{i,x_i}^1 \leftarrow \mathsf{Eval}(\mathsf{pp}, x, i)$ *and* $y_{i,x_i \oplus 1}^1 \xleftarrow{\$} \{0,1\}^\ell$*.*

In the full version we show how to instantiate robust hinting PRGs assuming the hardness of the CDH or the LWE problem.

5 A Universal TDFier

In the following we show how a generic compiler that takes as input any encryption scheme (that satisfies some mild structural properties) and makes it randomness recoverable, i.e. transforms it into a TDF. We call this scheme a *universal TDFier*.

5.1 One-Sided Predicate Encryption

We recall the notion of predicate encryption with one-sided security [23], which one of the most general derivations of the standard notion of public-key encryption.

Definition 8 (Predicate Encryption). *A family of one-sided predicate encryption schemes* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *consists of the following algorithms.*

- $\mathsf{Setup}(1^\lambda)$*: On input the security parameter, the setup algorithm returns a master secret key* msk *and some public parameters* pp*.*
- $\mathsf{KeyGen}(\mathsf{msk}, C)$*: On input the master secret key* msk *and a circuit* C*, the key generation algorithm returns a secret key* sk_C*.*
- $\mathsf{Enc}(\mathsf{pp}, \alpha, m)$*: On input the public parameters* pp*, an attribute* α*, and a message* m*, the evaluation algorithm returns a ciphertext* c*.*
- $\mathsf{Dec}(\mathsf{sk}_C, c)$*: On input a secret key* sk_C *for a circuit* C *and a ciphertext* c*, the decryption algorithm returns a message* m*.*

We require the following properties.

- *(Correctness) For all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{pp}, \mathsf{msk})$ *in the support of* $\mathsf{Setup}(1^\lambda)$*, all messages* m*, all attributes* α*, all circuits* C *such that* $C(\alpha) = 1$*, and all* sk_C *in the support of* $\mathsf{KeyGen}(\mathsf{msk}, C)$*, it holds that*

$$\mathsf{Dec}(\mathsf{sk}_C, \mathsf{Enc}(\mathsf{pp}, \alpha, m)) = m.$$

– *(One-Sided CPA Security)There exists a negligible function* negl *such that for all* $\lambda \in \mathbb{N}$*, all PPT adversaries* $\mathcal{A}$*, and all pairs of attributes* (α_0, α_1) *it holds that*

$$\Pr\left[\mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(c) = b \,\middle|\, \begin{array}{l}(\mathsf{pp},\mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ (m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{pp}) \\ b \xleftarrow{\$} \{0,1\} \\ c \xleftarrow{\$} \mathsf{Enc}(\mathsf{pp}, \alpha_b, m_b)\end{array}\right] - 1/2 = \mathsf{negl}(\lambda)$$

where the adversary never queries the KeyGen *oracle on some* C *such that* $C(\alpha_0) = 1$ *or* $C(\alpha_1) = 1$*.*

We remark that we require the scheme to satisfy perfect correctness, which is the case for most natural candidates of predicate encryption schemes (we refer the reader to the full version for a detailed discussion). We also note that generic transformation from approximate to perfect correctness are known [6].

Pseudorandom Ciphertexts. We additionally require that the ciphertext space satisfies some group-like structural properties. More specifically, we require the existence of a (not necessarily Abelian) group $\mathbb{H}$ with group operation $\circ$ such that (i) all ciphertexts in the range of the encryption algorithm consist of elements of $\mathbb{H}$ and (ii) undecryptable ciphertexts are computationally indistinguishable from uniformly sampled elements in $\mathbb{H}$. We define this more formally below.

Definition 9 (Pseudorandom Ciphertexts). *A one-sided predicate encryption scheme* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *has pseudorandom ciphertexts if there exists a negligible function* negl *and a group* $\mathbb{H}$ *such that for all* $\lambda \in \mathbb{N}$*, all PPT adversaries* $\mathcal{A}$*, and attributes* α *it holds that*

$$\Pr\left[\mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(c) = b \,\middle|\, \begin{array}{l}(\mathsf{pp},\mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ m \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{pp}) \\ b \xleftarrow{\$} \{0,1\} \\ c \xleftarrow{\$} \mathsf{Enc}(\mathsf{pp}, \alpha, m) \text{ if } b = 0 \\ c \xleftarrow{\$} \mathbb{H} \text{ if } b = 1\end{array}\right] - 1/2 = \mathsf{negl}(\lambda)$$

where the adversary never queries the KeyGen *oracle on some* C *such that* $C(\alpha) = 1$*.*

As an example, schemes that have ciphertexts indistinguishable from uniformly sampled bit strings satisfy this notion of security, since the set of all binary strings $\{0,1\}^p$ of length $p = p(\lambda)$ form a group with group operation $\oplus$. We will also consider schemes that have ciphertexts indistinguishable from uniformly sampled integers in $\mathbb{Z}_q^d$, with dimension $d = d(\lambda)$, where the group operation is the component-wise addition modulo q. Note that also a combination of both $\{0,1\}^p \times \mathbb{Z}_q^d$ satisfies this definition by defining the group operation canonically.

Selective vs Adaptive Security. As discussed before, we state the security definition in its selective variant, where the challenge attribute is fixed ahead of time. The definition can be extended to the adaptive settings canonically.

5.2 The Construction

In the following we present our compiler, which turns any one-sided predicate encryption scheme with pseudorandom ciphertexts into a predicate TDF for (k, n)-sources. The scheme is described below.

Construction 4 (Universal TDFier)**.** *Let* $(\mathsf{PE.Setup}, \mathsf{PE.KeyGen}, \mathsf{PE.Enc}, \mathsf{PE.Dec})$ *be a one-sided predicate encryption scheme with pseudorandom ciphertexts over* $\mathbb{H}$ *with group operation* $\circ$*, and let* $(\mathsf{HPRG.Setup}, \mathsf{HPRG.Eval})$ *be a* (k, n)*-robust hinting PRG. Our scheme* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Invert})$ *is defined as follows.*

- $\mathsf{Setup}(1^\lambda)$*: Invoke* $(\mathsf{pp}_{\mathsf{PE}}, \mathsf{msk}) \xleftarrow{\$} \mathsf{PE.Setup}(1^\lambda)$ *and* $\mathsf{pp}_{\mathsf{HPRG}} \xleftarrow{\$} \mathsf{HPRG.Setup}(1^\lambda)$*, then sample* $(r_1, \ldots, r_n) \xleftarrow{\$} \mathbb{H}^n$ *and a uniform* $u \xleftarrow{\$} \{0,1\}^{|C|+|\alpha|+3\lambda}$*. Set the public parameters of the scheme to be* $\mathsf{pp} := (\mathsf{pp}_{\mathsf{PE}}, \mathsf{pp}_{\mathsf{HPRG}}, r_1, \ldots, r_n, u)$ *and the master secret key to* msk*.*
- $\mathsf{KeyGen}(\mathsf{msk}, C)$*: Return the trapdoor* $\mathsf{td}_C \leftarrow \mathsf{PE.KeyGen}(\mathsf{msk}, C)$*.*
- $\mathsf{Eval}(\mathsf{pp}, \alpha, x)$*: On input some* $x \in \{0,1\}^n$*, for all* $i \in [n]$*, proceed as follows.*
  - *If* $x_i = 0$*: Compute*

$$d_i \leftarrow \mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \alpha, u; z_i)$$

    where $z_i \leftarrow \mathsf{HPRG.Eval}(\mathsf{pp}_{\mathsf{HPRG}}, x, i)$*. Set* $c_i := d_i$*.*
  - *If* $x_i = 1$*: Compute*

$$d_i \leftarrow \mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \alpha, u; z_i)$$

    where $z_i \leftarrow \mathsf{HPRG.Eval}(\mathsf{pp}_{\mathsf{HPRG}}, x, i)$*. Set* $c_i := d_i \circ r_i$*.*

  Return the image $y := (c_1, \ldots, c_n)$*.*
- $\mathsf{Invert}(\mathsf{td}_C, y)$*: On input some* $y \in \mathbb{H}^n$*, for all* $i \in [n]$*, proceed as follows: If* $u = \mathsf{Dec}(\mathsf{td}_C, c_i)$ *then set* $\tilde{x}_i := 0$*, else set* $\tilde{x}_i := 1$*. Return* $(\tilde{x}_1, \ldots, \tilde{x}_n)$*.*

Before proceeding with the analysis of our scheme, we highlight two important facts about the compiled TDF.

1. The public parameters of the scheme consist of the public parameters of the encryption scheme $\mathsf{pp}_{\mathsf{PE}}$, together with some independently sampled strings $(\mathsf{pp}_{\mathsf{HPRG}}, r_1, \ldots, r_n, u)$.
2. The master secret key and the user-specific keys are identical to those of the underlying encryption scheme.

Taken together, these imply that the underlying (predicate) encryption scheme can be upgraded to TDF (or, equivalently, made randomness recoverable) *after the fact*: Users of an existing (predicate) encryption scheme can decide to upgrade it to a TDF without the need to update their public nor their secret

keys. Instead they just need to add some public parameters (fixed once and for all) and modify their encryption/decryption procedure to achieve randomness recoverability. Alternatively, one can think of the above compiler as to add a *dual mode* to the encryption algorithm: Users are can choose whether they want to make their encryption randomness recoverable or not, without the need to change the public/secret keys of the scheme.

Correctness. We now show that the scheme as described above satisfies perfect correctness with all but negligible probability over the choice of the public parameters.

Theorem 5 (Correctness). *Let* $(\mathsf{PE.Setup}, \mathsf{PE.KeyGen}, \mathsf{PE.Enc}, \mathsf{PE.Dec})$ *be a one-sided predicate encryption with perfect correctness. Then Construction 4 satisfies correctness.*

Proof. We assume without loss of generality that the encryption algorithm of the one-sided predicate encryption scheme uses exactly λ-many bits of randomness. Recall that $\mathbb{H}$ is a bound on the ciphertext space of the scheme. Note that each secret key td_C defines a one-to-one mapping $\mathbb{H} \to P_C$, where the multiset P_C is populated by plaintexts (possibly with repeated elements). Define P to be the multiset that contains (possibly with repeated elements) all P_i for $i \in \left[2^{|C|}\right]$, where $|P| = |\mathbb{H}| \cdot 2^{|C|}$. Let $S_u \subseteq P$ be the subset of P where all entries of S_u are equal to u. In expectation, over the random choice of u, we have that

$$\mathsf{E}\left[|S_u|\right] = \frac{|P|}{2^{|u|}} = \frac{|\mathbb{H}| \cdot 2^{|C|}}{2^{|u|}}.$$

By Lemma 1, we have that

$$|S_u| \leq \frac{|\mathbb{H}| \cdot 2^{|C|} \cdot 2^{\lambda}}{2^{|u|}}$$

except with probability $2^{-\lambda}$, over the random choice of u. Define $T_u \subseteq \mathbb{H}$ to be the set of all pre-images of S_u. Note that

$$|T_u| \leq |S_u| \leq \frac{|\mathbb{H}| \cdot 2^{|C|} \cdot 2^{\lambda}}{2^{|u|}}.$$

Let R_u be the set such that for all $r \in R_u$ there exist ciphertexts $\mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \cdot, u)$ such that $\mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \cdot, u) \circ r \in T_u$. Note that there are at most $2^{|\alpha|} \cdot 2^{\lambda}$ many ciphertexts in the support of $\mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \cdot, u)$ and therefore we can bound

$$|R_u| \leq \frac{|\mathbb{H}| \cdot 2^{|C|} \cdot 2^{\lambda} \cdot 2^{|\alpha|} \cdot 2^{\lambda}}{2^{|u|}} = \frac{|\mathbb{H}| \cdot 2^{|C|+|\alpha|+2\lambda}}{2^{|u|}}$$

by a counting argument. Thus the probability that a uniformly chosen $r \xleftarrow{\$} \mathbb{H}$ belongs to R_u is at most

$$\Pr[r \in R_u] \leq \frac{|\mathbb{H}| \cdot 2^{|C|+|\alpha|+2\lambda}}{|\mathbb{H}| \cdot 2^{|u|}} = \frac{2^{|C|+|\alpha|+2\lambda}}{2^{|C|+|\alpha|+3\lambda}} = \frac{1}{2^{\lambda}}.$$

Note that a decryption error only happens whenever some r_i maps a valid encryption of u to some other encryption of u, i.e. $r \in R_u$. By a union bound, the probability that at least one of the elements $(r_1, \ldots, r_n)$ belong to such a set is also negligible. This concludes our proof. □

Security. We now turn to prove the deterministic CPA security of our scheme. Note that we restrict our analysis to the selective variant of the security definition.

Theorem 6 (CPA Security). *Let* $(\mathsf{PE.Setup}, \mathsf{PE.KeyGen}, \mathsf{PE.Enc}, \mathsf{PE.Dec})$ *be a one-sided predicate encryption scheme with pseudorandom ciphertexts over* $\mathbb{H}$ *with group operation* $\circ$*, and let* $(\mathsf{HPRG.Setup}, \mathsf{HPRG.Eval})$ *be a* (k, n)*-robust hinting PRG. Then Construction 4 satisfies selective* (k, n)*-CPA security.*

Proof. The proof proceeds by a series on hybrids where we gradually change the distribution of the public parameters and of the challenge ciphertext.

- Hybrid $\mathcal{H}_0$: This is the original experiment with the challenge bit b fixed to 0.
- Hybrid $\mathcal{H}_1$: In this hybrid we first compute the challenge ciphertext and then we set the values of $(r_1, \ldots, r_n)$ accordingly. More specifically, for all $i \in [n]$, we do the following:
  - If $x_i = 0$ compute $c_i \leftarrow \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha, u; z_i)$, then define $r_i := c_i \circ s_i$, where $s_i \xleftarrow{\$} \mathbb{H}$.
  - If $x_i = 1$ compute $c_i \xleftarrow{\$} \mathbb{H}$ and define $r_i := c_i \circ \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u; z_i)$.

  Since $\mathbb{H}$ is a group and in particular all elements of $\mathbb{H}$ have an inverse, the distribution induced by this hybrid is identical to the previous one and thus the change described here is only syntactical.
- Hybrids $\mathcal{H}_2 \ldots \mathcal{H}_{n+1}$: For all $i \in [2, n+1]$ we define $\mathcal{H}_i$ as the previous one, except for the following modification:
  - If $x_i = 0$ compute $c_i \leftarrow \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u; z_i)$, then define $r_i := c_i \circ s_i$, where $s_i \xleftarrow{\$} \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u)$.
  - If $x_i = 1$ compute $c_i \xleftarrow{\$} \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u)$, then define the variable $r_i := c_i \circ \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u; z_i)$.

  I.e. instead of sampling a random mask, we compute an encryption of u under the appropriate attribute using fresh random coins. Indistinguishability follows from a routine reduction against the (selective) pseudorandom ciphertexts of the one-sided predicate encryption scheme.
- Hybrid $\mathcal{H}_{n+2}$: In this hybrid we compute the challenge ciphertext and the setup using fresh random coins. More precisely, for all $i \in [n]$ we do the following:
  - If $x_i = 0$ compute $c_i \xleftarrow{\$} \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u)$, then define $r_i := c_i \circ s_i$, where $s_i \xleftarrow{\$} \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u)$.
  - If $x_i = 1$ compute $c_i \xleftarrow{\$} \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u)$, then define $r_i := c_i \circ s_i$, where $s_i \xleftarrow{\$} \mathsf{PE.Enc}(\mathsf{pp_{PE}}, \alpha_0, u)$.

Note that the only difference between this hybrid and the previous one is that we use fresh coins instead of pseudorandom ones derived from applying the hinting PRG to x_0. Furthermore, note that the only information about x is encoded in the positions where we used truly random coins instead of pseudorandom. Since x_0 is a (k, n)-source, indistinguishability follows by a reduction against the (k, n)-robustness of the hinting PRG.

- Hybrid $\mathcal{H}_{n+3}$: In this hybrid we fix the challenge bit b to 1. Note that the challenge ciphertext does no longer depend on the input x so the only difference here is that we compute all ciphertexts as $\mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \alpha_1, u)$ instead of $\mathsf{PE.Enc}(\mathsf{pp}_{\mathsf{PE}}, \alpha_0, u)$. Indistinguishability follows by a standard hybrid argument against the one-sided CPA security of the predicate encryption scheme.
- Hybrids $\mathcal{H}_{n+4} \ldots \mathcal{H}_{2n+5}$: In these hybrids we undo the changes that we performed in the previous hybrids, except for switching the challenge bit. The indistinguishability arguments are identical. The final hybrid is the original experiment with b fixed to 1.

To summarize, we have that

$$\mathcal{H}_0 \equiv \mathcal{H}_1 \stackrel{c}{\equiv} \mathcal{H}_2 \stackrel{c}{\equiv} \ldots \stackrel{c}{\equiv} \mathcal{H}_{n+1} \stackrel{c}{\equiv} \mathcal{H}_{n+2} \stackrel{c}{\equiv} \mathcal{H}_{n+3} \stackrel{c}{\equiv} \mathcal{H}_{n+4} \stackrel{c}{\equiv} \ldots \stackrel{c}{\equiv} \mathcal{H}_{2n+4} \equiv \mathcal{H}_{2n+5}$$

which implies that the scheme is selective (k, n)-CPA secure. □

6 Trapdoor Garbled Circuits and Adaptive Single-Key AB-TDFs

The AB-TDF scheme in Section 5 is only selectively secure. In this section we show an alternative way of building single-key AB-TDFs, that provides adaptive security in the choice of the attribute α (i.e., α can be chosen adaptively based on the ABE public parameter). Along the way, we define a concept called trapdoor garbling, and show how to build it from DDH/LWE.

Adaptive Security for Single-Key AB-TDF. For simplicity, we will focus on one-wayness only (as opposed to deterministic-encryption security). Here the adversary after seeing pp can choose a circuit C and an attribute α satisfying $C(\alpha) = 0$, and is tasked with recovering a random $x \stackrel{\$}{\leftarrow} \{0,1\}^n$ from sk_C and $y := \mathsf{Eval}(\mathsf{pp}, \alpha, x)$.

6.1 Definition

Garbled circuits in a traditional sense allows one to garble a circuit $\mathsf{P} : \{0,1\}^m \to \{0,1\}$, so that a garbled circuit and a corresponding garbled label for an input x reveals nothing beyond $\mathsf{P}(x)$ — in particular, the randomness used by the garbling algorithm as well as any possible circuit-hardcoded information should remain hidden. We introduce and realize a notion of garbled circuits which allows one to recover the randomness on specific garbled inputs. We define it for the *universal circuit* below.

Definition 10 (Trapdoor Garbling). *Let* $\mathsf{U}[\cdot,\cdot]$ *be a circuit that works as follows: the output of* $\mathsf{U}[s_1,s_2](C)$ *is* s_2 *if* $C(s_1)=1$*, and is a special symbol* \$ *otherwise. We define a trapdoor garbling scheme* $\mathsf{GRB}=(\mathsf{Garble},\mathsf{Eval},\mathsf{Sim})$ *for* $\mathsf{U}[\cdot,\cdot]\colon\{0,1\}^m\to\{0,1\}^\kappa\cup\{\$\}$*.*

- $\mathsf{Garble}(1^\lambda,s_1,s_2;\rho)$*: On input the security parameter* 1^λ *and private hard-coded strings* s_1,s_2*, and randomness* ρ*, the garbling algorithm returns a garbled circuit* $\tilde{\mathsf{U}}$ *and a set of* m *pairs of labels* $\{\ell_{i,0},\ell_{i,1}\}_{i\in[m]}$*.*
- $\mathsf{Eval}(\tilde{\mathsf{U}},\{\ell_{i,C_i}\}_{i\in[m]})$*: On input a garbled circuit* $\tilde{\mathsf{U}}$ *and a set of labels* $\{\ell_{i,C_i}\}_{i\in[m]}$*, the evaluation algorithm returns an output string* $y\in\{0,1\}^\kappa\cup\{\$\}$*.*

We require the following properties.

- *(Correctness) For all* $\lambda\in\mathbb{N}$*, all* s_1*, all* $s_2\in\{0,1\}^\kappa$*,* $C\in\{0,1\}^m$*, and garbling randomness* ρ*, letting* $(\tilde{\mathsf{U}},\{\ell_{i,0},\ell_{i,1}\}_{i\in[m]})\xleftarrow{\$}\mathsf{Garble}(1^\lambda,s_1,s_2;\rho)$*:*
  - *if* $\mathsf{U}[s_1,s_2](C)=\$$*, then* $\mathsf{Eval}\left(\tilde{\mathsf{U}},\{\ell_{i,x_i}\}_{i\in[m]}\right)=\$$
  - *else,* $\mathsf{Eval}\left(\tilde{\mathsf{U}},\{\ell_{i,C_i}\}_{i\in[m]}\right)=(s_2,\rho)$*.*
- *(Simulation Security) For any "admissible" PPT adversary* $\mathcal{A}$*, the following holds. Letting* $((s_1,s_2,C),\mathsf{st})\xleftarrow{\$}\mathcal{A}(1^\lambda)$*,* $\rho\xleftarrow{\$}\{0,1\}^*$*, and* $(\tilde{\mathsf{U}},\{\ell_{i,0},\ell_{i,1}\}_{i\in[m]})=\mathsf{Garble}(1^\lambda,s_1,s_2;\rho)$*:*

$$\left(\mathsf{st},\tilde{\mathsf{U}},\{\ell_{i,C_i}\}_{i\in[m]}\right)\stackrel{c}{\equiv}\left(\mathsf{st},\mathsf{Sim}\left(1^\lambda,|s_1|,|s_2|,\mathsf{U}[s_1,s_2](C)\right)\right).$$

  We say $\mathcal{A}$ *is admissible if* $\mathsf{U}[s_1,s_2](C)=\$$*, where all strings are as above.*

6.2 Tools for Building Single-Key AB-TDFs

We show how to build single-key AB-TDFs from DDH/LWE. (All our results will also apply to the predicate-encryption setting.) Our techniques will implicitly also realize trapdoor garbling (Definition 10).

Definition 11 (Randomness Recoverable SKE). *We say* $\mathsf{SKE}:=(\mathsf{G},\mathsf{Enc},\mathsf{Dec})$ *is randomness recoverable if (1) a key is chosen at random from* $\{0,1\}^n$ *for some* $n=n(\lambda)$*; and (2) given* k *and* $\mathsf{Enc}(k,m;r)$ *we can recover both* m *and* r*.*

Notation. We use the shorthand $\{a_{i,b}\}$ to mean $\{a_{i,b}\}_{i\in[m],b\in\{0,1\}}$.

We will now define and later realize an enhanced version of Yao's garbled circuits. Informally, this enhancement allows the recovery of the garbled circuits coins, in trapdoor mode (i.e., when we have labels corresponding to an input which makes the circuit evaluate to one). We explained the high-level idea in the introduction, and will now formalize it.

In the construction below, we assume the following for Yao's scheme for garbling single-bit output circuits:

Construction 7 (Enhanced Garbled Circuits)**.** *We describe an enhanced way of garbling* $\mathsf{U}[\cdot,\cdot]$*, introduced in Definition 10. Let* $\mathsf{P}[\cdot]$ *be a circuit, where for* $\alpha \in \{0,1\}^k$ *and* $C \in \{0,1\}^m$*,* $\mathsf{P}[\alpha](C) = C(\alpha)$*. Let* $(\mathsf{Garble}, \mathsf{Eval})$ *be Yao's garbled-circuit scheme for* $\mathsf{P}[\cdot]$*, as described in [37, 30], with the following slight modification: Letting* $k_{\mathsf{out},0}$ *and* $k_{\mathsf{out},1}$ *be the keys for the two values of the output wire, instead of appending* $(k_{\mathsf{out},0}, 0)$ *and* $(k_{\mathsf{out},1}, 1)$ *to the garbled circuit, we append* $\mathsf{Enc}(k_{\mathsf{out},0}, 0)$ *and* $\mathsf{Enc}(k_{\mathsf{out},1}, 1)$ *to the garbled circuit (i.e., the values of* $k_{\mathsf{out},0}$ *and* $k_{\mathsf{out},1}$ *are not copied in the garbled circuit in the clear).*

- $\overline{\mathsf{Garble}}(\alpha, x)$*: Sample* $(\tilde{C}, \{\mathsf{lb}_{i,b}\}) \xleftarrow{\$} \mathsf{Garble}(\mathsf{P}, \alpha)$ *and let* $k_{\mathsf{out},1}$ *be output wire for bit value* 1*. Let* Key *be the set of all keys for the circuit wires (i.e., two keys for each wire), and let* $\mathsf{CT} = \{\mathsf{Enc}(k_{\mathsf{out},1}, x)\} \cup \{\mathsf{Enc}(k_{\mathsf{out},1}, k) \colon k \in \mathsf{Key} \setminus \{k_{\mathsf{out},1}\}\}$*. Let* $\widetilde{C_{\mathsf{en}}} := (\mathsf{CT}, \tilde{C})$*, and return* $(\widetilde{C_{\mathsf{en}}}, \{\mathsf{lb}_{i,b}\}_{i\in[m],b\in\{0,1\}})$*.*
- $\overline{\mathsf{Eval}}(\widetilde{C_{\mathsf{en}}}, \mathbf{lb})$*: Parse* $\widetilde{C_{\mathsf{en}}} := (\mathsf{CT}, \tilde{C})$*. Let* $b := \mathsf{Eval}(\tilde{C}, \mathbf{lb})$*. If* $b = 0$*, return* $\$$*; otherwise, letting* $k_{\mathsf{out},1}$ *be the key for the output wire, return* $(\mathsf{Dec}(k_{\mathsf{out},1}, \mathsf{CT}))$*, where* $\mathsf{Dec}(k_{\mathsf{out},1}, \mathsf{CT}) = \{\mathsf{Dec}(k_{\mathsf{out},1}, c) \colon c \in \mathsf{CT}\}$*.*

The following lemma shows that given an enhanced garbled circuit and a sequence of accepting labels, then in addition to x, we can recover the randomness used to garble the circuit.

Lemma 3 (Randomness Recoverability of the Enhanced Garbled Circuit). *Suppose* $\mathsf{SKE} := (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ *is randomness recoverable. Fix randomness* ρ *for* $\overline{\mathsf{Eval}}$ *and let* $(\widetilde{C_{\mathsf{en}}}, \{\mathsf{lb}_{i,b}\}) := \overline{\mathsf{Garble}}(\alpha, x; \rho)$*. Assuming* $\overline{\mathsf{Eval}}(\widetilde{C_{\mathsf{en}}}, \{\mathsf{lb}_{i,C_i}\}) = (x, \omega)$*, then given* ω *we can recover the original randomness* ρ*.*

Proof. Notice that ρ consist of two sources of randomness: (1) those used to generate the keys for the wires (i.e., the keys in set Key, Construction 7) and (2) the random coins used to encrypt the keys. Given $k_{\mathsf{out},1}$, we can recover all the keys in Key (since they are all encrypted under $k_{\mathsf{out},1}$) and hence by Definition 11 all random coins involving Source (1) are recovered. Having recovered all the keys in Key, by Definition 11 we can recover all the coins used to generate the ciphertexts. □

6.3 Single-Key AB-TDFs Construction

We now give the construction.

Construction 8 (Single-Key AB-TDF)**.** *Let* m *and* k *be the size of the circuit and attribute.*

Ingredients. *A randomness recoverable secret-key encryption scheme* $\mathsf{SKE} = (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ *(Definition 11), a PKE scheme* $\mathsf{PKE} = (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ *and an enhanced Yao's garbling scheme* $(\overline{\mathsf{Garble}}, \overline{\mathsf{Eval}})$ *(Construction 7).*

Input to the TDF. *The input to a function is of the form* $(\mathbf{r}, \rho)$*, consisting of* $2n$ *randomness values* $\mathbf{r} := \{r_{i,b}\}$ *for* $\mathsf{PKE.Enc}$ *and a randomness string* ρ *for the garble function* $\overline{\mathsf{Garble}}(\mathsf{U}[\alpha, \mathbf{r}])$*.*

- $\mathsf{Setup}(1^\lambda)$: *for* $i \in [m]$ *and* $b \in \{0,1\}$: $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b}) \xleftarrow{\$} \mathsf{PKE.G}(1^\lambda)$. *Let* $\mathsf{pp} := \{\mathsf{pk}_{i,b}\}_{i\in[m],b\in\{0,1\}}$ *and* $\mathsf{msk} := \{\mathsf{sk}_{i,b}\}_{i\in[m],b\in\{0,1\}}$.
- $\mathsf{KeyGen}(\mathsf{msk}, C)$: *output* $\mathsf{sk}_C := (C, \mathsf{sk}_{1,C_1}, \ldots, \mathsf{sk}_{m,C_m})$.
- $\mathsf{Eval}(\mathsf{pp}, \alpha, x)$: *parse* $x := (\mathbf{r} := \{r_{i,b}\}_{i\in[m],b\in\{0,1\}}, \rho)$. *Let* $(\tilde{C}, \mathbf{lb}) = \overline{\mathsf{Garble}}(\alpha, x; \rho)$, *and parse* $\mathbf{lb} := \{\mathsf{lb}_{i,b}\}_{i\in[m],b\in\{0,1\}}$. *Let* $\mathsf{ct}_{i,b} := \mathsf{PKE.Enc}(\mathsf{pk}_{i,b}, \mathsf{lb}_{i,b}; r_{i,b})$ *and return* $y := (\tilde{C}, \{\mathsf{ct}_{i,b}\})$.
- $\mathsf{Invert}(\mathsf{sk}_C, y)$: *Parse* $\mathsf{sk}_C := (C, \tilde{\mathsf{sk}}_1, \ldots, \tilde{\mathsf{sk}}_m)$ *and* $y := (\tilde{C}, \tilde{\mathsf{ct}}_1, \ldots, \tilde{\mathsf{ct}}_m)$ *and let* $\ell_i := \mathsf{PKE.Dec}(\tilde{\mathsf{sk}}_i, \tilde{\mathsf{ct}}_i)$ *for* $i \in [m]$. *Run* $\overline{\mathsf{Eval}}(\tilde{C}, \{\ell_i\})$; *if the output is* \$, *return* $\perp$; *otherwise, parsing the output* (x, ω), *return* (x, ρ) *where* ρ *is computed from* ω *as shown in Lemma 3.*

Lemma 4 (Correctness). *Assuming* $\mathsf{SKE} := (\mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$ *is randomness recoverable (Definition 11), the resulting scheme PE-TDF in Construction 8 has perfect correctness (Definition 3).*

Proof. The proof follows because by Lemma 3 the enhanced version of Yao's garbled circuit is randomness recoverable (hence recovering ρ), and also all the random coins used to encrypt the labels (i.e., $\mathbf{r}$ in Construction 8) are outputted by the evaluation algorithm on an accepting sequence of garbled labels. □

Instantiating the Encryption Schemes in Construction 8 Construction 8 introduces a circularity: the labels of the garbled circuit are encrypted under a PKE scheme using randomness $\mathbf{r}$, and the underlying randomness $\mathbf{r}$ is hardwired into the circuit being garbled. We now show how to overcome this circularity in a provable way using the following instantiations: the underlying PKE scheme will be Dual-BHHO (which we call $\mathsf{PKE}_{\mathrm{dbhho}}$), while the secret-key encryption scheme is BHHO, adapted to the private-key setting.

Construction 9 (Private-Key BHHO)**.** *Define* $\mathsf{SKE} = (\mathsf{SKE.G}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ *as follows:*

- $\mathsf{SKE.G}(1^\lambda)$: *return* $s \xleftarrow{\$} \{0,1\}^n$.
- $\mathsf{SKE.Enc}(s, g')$: *sample* $(g_1, \ldots, g_n) \xleftarrow{\$} \mathbb{G}$ *and return* $(g_1, \ldots, g_n, g' \times \Pi g_i^{s_i})$.
- $\mathsf{SKE.Dec}(s, \mathsf{ct})$: *parse* $\mathsf{ct} := (\mathbf{g}, g'')$; *return,* $g''/(s \cdot \mathbf{g})$.

Randomness Recoverability. For SKE of Construction 9, the encryption algorithm also samples n group elements, and one might complaint that we cannot necessarily recover the underlying coins used to generate these group elements. However, this fact can be handled by putting all these group elements in the public parameter pp of the TD-ABE scheme (Construction 8). In other words, the algorithm Setup of Construction 8 will include n group elements in pp for every private-key encryption that is going to be performed during Eval (more specifically, during $\overline{\mathsf{Garble}}$). We ignore this fact here, and we hereon assume these group elements are generated as part of each encryption.

We review the scheme of k-block BHHO, which outputs k ciphertexts each sampled under BHHO.

Construction 10 (k-block BHHO [10])**.** *We review the definition of the BHHO scheme* $\mathsf{PKE}_{\mathrm{bhho}}$ *for encrypting* k *group elements.*

- $\mathsf{S}(1^\lambda)$*: Sample* n *random group elements* $\mathsf{pp} := \mathbf{g} := (g_1, \ldots, g_n)$*, where* $n \in \omega(\log p)$*, where* $p = |\mathbb{G}|$*.*
- $\mathsf{G}(\mathsf{pp})$*: On* $\mathsf{pp} := \mathbf{g}$*, return* $(\mathsf{pk}, \mathsf{sk} := s)$*, where* $s \xleftarrow{\$} \{0,1\}^n$ *and* $\mathsf{pk} := (\mathbf{g}, s \cdot \mathbf{g})$*. By abusing notation, we may sometimes write* $\mathsf{pk} = \mathsf{G}(\mathsf{pp}, \mathsf{sk})$*.*
- $\mathsf{Enc}(\mathsf{pk}, \mathbf{m} := (g'_1, \ldots, g'_k))$*: To encrypt* $\mathbf{m}$ *under* $\mathsf{pk} := (\mathbf{g}, g_{\mathsf{pk}})$*, sample a* k*-tuple randomness* $(r_1, \ldots, r_k) \xleftarrow{\$} \mathbb{Z}_p^k$ *and return* $\mathsf{ct} := (\mathbf{g}^{r_1}, g_{\mathsf{pk}}^{r_1} \cdot g'_1, \ldots, \mathbf{g}^{r_k}, g_{\mathsf{pk}}^{r_k} \cdot g'_k)$*.*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$*: Obvious.*

Construction 11 (k-block Dual BHHO)**.** *Define* $\mathsf{PKE}_{\mathrm{dbhho}} = (\mathsf{S}, \mathsf{G}, \mathsf{Enc}, \mathsf{Dec})$*, the* k*-block version of* DualBHHO, as follows.

- $\mathsf{S}(1^\lambda)$: Sample n random group elements $\mathsf{pp} := \mathbf{g} := (g_1, \ldots, g_n)$, where $n \in \omega(\log(|\mathbb{G}|))$.
- $\mathsf{G}(\mathbf{g})$: On $\mathsf{pp} := \mathbf{g}$, return $(\mathsf{pk}, \mathsf{sk})$, where $\mathsf{sk} := (r_1, \ldots, r_k) \xleftarrow{\$} \mathbb{Z}_p^k$ and $\mathsf{pk} := (\mathbf{g}, \mathbf{g}^{r_1}, \ldots, \mathbf{g}^{r_k})$. By abusing notation, we may sometimes write $\mathsf{pk} = \mathsf{G}(\mathsf{pp}, \mathsf{sk})$.
- $\mathsf{Enc}(\mathsf{pk}, g_m)$: To encrypt k group element $(g_1, \ldots, g_k)$ under $\mathsf{pk} := (\mathbf{g}, \mathbf{g}_1, \ldots, \mathbf{g}_k)$, sample randomness $s \xleftarrow{\$} \{0,1\}^n$ and return $\mathsf{ct} := (s \cdot \mathbf{g}, (s \cdot \mathbf{g}_1) \times g_1, \ldots, (s \cdot \mathbf{g}_k) \times g_k)$, where $s \cdot \mathbf{g} := \Pi g_i^{s_i}$.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: parse $\mathsf{sk} := (r_1, \ldots, r_k)$ and $\mathsf{ct} := (g_1, g_2, \ldots, g_{k+1})$, then return $(g_2/(g_1)^{r_1}, \ldots, g_{k+1}/(g_1)^{r_k})$.

We now prove a technique for switching RDM+KDM security to KDM security. Recall that under $\mathsf{PKE}_{\mathrm{dbhho}}$ the encryption randomness is a string $s \in \{0,1\}^n$, the same as a secret key for $\mathsf{PKE}_{\mathrm{bhho}}$. Similarly, a secret key $(r_1, \ldots, r_k)$ under $\mathsf{PKE}_{\mathrm{dbhho}}$ corresponds to encryption randomness for k-block $\mathsf{PKE}_{\mathrm{bhho}}$. We give the following lemma, and will then discuss its usefulness.

Lemma 5 (RDM/KDM Switching Lemma). *Let* $\mathbf{m} = (m_1, \ldots, m_k)$ *be a sequence of* k *group elements, to be encrypted. Fix a public parameter* $\mathbf{g} := (g_1, \ldots, g_n)$ *across both* $\mathsf{PKE}_{\mathrm{bhho}}$ *and* $\mathsf{PKE}_{\mathrm{dbhho}}$*. Let*

- $\mathbf{r} := (r_1, \ldots, r_k) \in \mathbb{Z}_p^k$ *be a secret key under* $\mathsf{PKE}_{\mathrm{dbhho}}$ *and let* pk *be the corresponding public key;* $\mathsf{pk} := (\mathbf{g}, \mathbf{g}^{r_1}, \ldots, \mathbf{g}^{r_k})$*.*
- $s \in \{0,1\}^n$ *be a randomness value under* $\mathsf{PKE}_{\mathrm{dbhho}}$*. Also, let* pk' *be the corresponding BHHO public key under* s*:That is,* $\mathsf{pk}' := (\mathbf{g}, s \cdot \mathbf{g})$*.*

Up to "rearrangement of the terms":

$$\mathsf{pk}, \mathsf{Enc}_{\mathrm{dbhho}}(\mathsf{pk}, \mathbf{m}; s) = \mathsf{pk}', \mathsf{Enc}_{\mathrm{bhho}}(\mathsf{pk}', \mathbf{m}; \mathbf{r}),$$

where $=$ *indicates that the two distributions are identical.*

Usefulness of Lemma 5. Let SKE be the private-key BHHO scheme (Construction 9) and assume $|s| = n$. Then we can reduce a combination of RDM and KDM attacks into a solely KDM scenario.

$$\underbrace{\mathsf{pk}, \mathsf{Enc}_{\mathrm{dbhho}}(\mathsf{pk}, s; s'), \mathsf{SKE.Enc}(s, s')}_{\text{RDM+KDM}} \equiv \underbrace{\mathsf{pk}', \mathsf{Enc}_{\mathrm{bhho}}(\mathsf{pk}', s; \mathbf{r}), \mathsf{SKE.Enc}(s, s')}_{\text{KDM Only}} \tag{4}$$

where $s \xleftarrow{\$} \{0,1\}^n$, $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^n$, $\mathsf{pk} := \mathsf{PKE}_{\mathrm{dbhho}}.\mathsf{G}(\mathsf{pp}, \mathbf{r})$, $s' \xleftarrow{\$} \{0,1\}^n$ and $\mathsf{pk}' := \mathsf{G}_{\mathrm{bhho}}(\mathsf{pp}, s')$. Notice that the randomness $\mathbf{r}$ in the righthand side is not used anywhere else in that side, so we do not have randomness dependency anymore.

Proof of Lemma 5. The proof follows easily by inspection. Letting $\mathbf{g} := (g_1, \ldots, g_n)$:

$$\mathsf{pk}, \mathsf{Enc}_{\mathrm{dbhho}}(\mathsf{pk}, \mathbf{m}; s) := \begin{pmatrix} g_1, g_2, \ldots, g_n \\ g_1^{r_1}, g_2^{r_1}, \ldots, g_n^{r_1} \\ \vdots \\ g_1^{r_k}, g_2^{r_k}, \ldots, g_n^{r_k} \end{pmatrix}, \begin{pmatrix} s \cdot \mathbf{g} \\ (s \cdot \mathbf{g})^{r_1} \cdot m_1 \\ \vdots \\ (s \cdot \mathbf{g})^{r_k} \cdot m_k \end{pmatrix}.$$

Thus, we may concisely write

$$\mathsf{pk}, \mathsf{Enc}_{\mathrm{dbhho}}(\mathsf{pk}, \mathbf{m}; s) := \begin{pmatrix} g_1, & g_2, \ldots, & g_n, & s \cdot \mathbf{g} \\ g_1^{r_1}, & g_2^{r_1}, \ldots, & g_n^{r_1}, & (s \cdot \mathbf{g})^{r_1} \cdot m_1 \\ & & \vdots & \\ g_1^{r_k}, & g_2^{r_k}, \ldots, & g_n^{r_k}, & (s \cdot \mathbf{g})^{r_k} \cdot m_k \end{pmatrix}, \tag{5}$$

Recall that $\mathsf{pk}' = (g_1, \ldots, g_n, s \cdot \mathbf{g})$, which is the first column of the matrix in Equation 5. Thus, the matrix in Equation 5 corresponds to $\mathsf{pk}', \mathsf{Enc}_{\mathrm{bhho}}(\mathsf{pk}', \mathbf{m}; \mathbf{r})$, up to obvious rearrangement of the terms.

□

Lemma 6 (Adaptive Security for AB-TDF). *Assuming the DDH assumption holds. Instantiating Construction with* SKE *of Construction 9 and* PKE *of Construction 11 and an enhanced garbled circuit (Construction 7), the AB-TDF scheme of Construction 8 is single-key adaptively secure.*

Proof. Let $(\tilde{C}, \{\mathsf{lb}_{i,b}\})$ be the resulting garbled circuits and labels. Let

- $k_{1,0}, k_{1,1}, \ldots, k_{m,0}, k_{m,1}$ be the input labels of the garbled circuit (i.e., $\mathsf{lb}_{i,b} = k_{i,b}$), and let $\{\mathsf{pk}_{i,b}\}$ be the $2n$ pairs of public keys and secret keys used to encrypt the corresponding input label.
- Key be the set of all keys sampled during the garbled circuit construction (Construction 7).

Also, let $k_{\mathsf{out},0}$ and $k_{\mathsf{out},1}$ be the output-wire keys corresponding to bit values zero and one. Let $\mathsf{CT}_{\mathsf{all}}$ be the set of all ciphertexts in the image y. We may split $\mathsf{CT}_{\mathsf{all}}$ into three subsets:

1. $\mathsf{CT}_{\mathsf{all}1}$: label encryptions:

$$\mathsf{CT}_{\mathsf{all}1} : \{\mathsf{PKE}_{\mathrm{dbhho}}.\mathsf{Enc}(\mathsf{pk}_{i,b}, k_{i,b}; r_{i,b}) \colon i \in [m], b \in \{0,1\}\};$$

2. $\mathsf{CT}_{\mathsf{all}2}$: encryptions of the random coins $\{r_{i,b}\}$ used in Step 1 as well as the garbled circuit keys Key, made under $k_{\mathsf{out},1}$:

$$\mathsf{CT}_{\mathsf{all}2} := \{\mathsf{SKE}.\mathsf{Enc}(k_{\mathsf{out},1}, r_{i,b}) \colon i \in [m], b \in \{0,1\}\} \cup \{\mathsf{SKE}.\mathsf{Enc}(k_{\mathsf{out},1}, k) \colon k \in \mathsf{Key} \setminus \{k_{\mathsf{out},1}\}\}$$

   where all encryptions in $\mathsf{CT}_{\mathsf{all}2}$ use fresh randomness.
3. $\mathsf{CT}_{\mathsf{all}3}$: all intermediate key encryptions, as per Yao's garbled circuit construction(Construction 7).

Notice the RDM/KDM circularity involved between $\mathsf{CT}_{\mathsf{all}1}$ and $\mathsf{CT}_{\mathsf{all}2}$: $\mathsf{pk}_{i,b}$ are encrypting $k_{i,b}$ (which in turn encrypt $k_{\mathsf{out},1}$ via a sequence of hops), and the random coins used to encrypt $k_{i,b}$ under $\mathsf{pk}_{i,b}$ are encrypted under $k_{\mathsf{out},1}$.

We will now use Lemma 5 to reduce the above RDM+KDM dependency to KDM-dependency alone, at which point we can use the BHHO result to argue security for the garbled circuit.

Let $\mathbf{h}_{i,b} \xleftarrow{\$} \mathbb{Z}_p^k$ be the randomness used to generate $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b})$, and note that this randomness is never encrypted in CT. Also, recall that $r_{i,b} \xleftarrow{\$} \{0,1\}^n$. By Lemma 5

$$\mathsf{PKE}_{\mathrm{dbhho}}.\mathsf{Enc}(\mathsf{pk}_{i,b}, k_{i,b}; r_{i,b}) := \mathsf{PKE}_{\mathrm{bhho}}.\mathsf{Enc}(\mathsf{pk}'_{i,b}, k_{i,b}; \mathbf{h}_{i,b}), \tag{6}$$

where $\mathsf{pk}'_{i,b} = \mathsf{PKE}_{\mathrm{bhho}}.\mathsf{G}(\mathsf{pp}, r_{i,b})$. In other words, $r_{i,b}$ is now the secret key of $\mathsf{pk}'_{i,b}$. With this in mind, we may write

$$\mathsf{CT}_{\mathsf{all}1} : \{\mathsf{PKE}_{\mathrm{bhho}}.\mathsf{Enc}(\mathsf{pk}'_{i,b}, k_{i,b}; \mathbf{h}_{i,b}) \colon i \in [m], b \in \{0,1\}\}. \tag{7}$$

Notice that $\mathsf{CT}_{\mathsf{all}2}$ is now encrypting the secret keys of $\mathsf{pk}'_{i,b}$, and thus we are in a KDM-only scenario.

Once having reduced RMD/KDM to KDM-only in the garbled circuits, the rest of the proof follows as in [37, 30]. □

Instantiation Using LWE. Instantiating Construction 8 with SKE which is dual-Regev's circularly-secure SKE scheme [22], with PKE which is Regev's PKE scheme [34] and an enhanced garbled circuit (Construction 7) based on SKE above, the AB-TDF scheme of Construction 8 is adaptively secure. The proof will be the same, since we can prove the RDM/KDM switching lemma (Lemma 5) based on these encryption schemes. See the full version for further details.

References

1. Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In Ran Canetti

and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 57–74, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany. 4

2. Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany. 3
3. Mihir Bellare, Zvika Brakerski, Moni Naor, Thomas Ristenpart, Gil Segev, Hovav Shacham, and Scott Yilek. Hedged public-key encryption: How to protect against bad randomness. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 232–249, Tokyo, Japan, December 6–10, 2009. Springer, Heidelberg, Germany. 2, 3
4. Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany. 2
5. Mihir Bellare, Eike Kiltz, Chris Peikert, and Brent Waters. Identity-based (lossy) trapdoor functions and applications. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 228–245, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. 3, 4, 14
6. Nir Bitansky and Vinod Vaikuntanathan. A note on perfect correctness by derandomization. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 592–606, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany. 17
7. Alexandra Boldyreva, Serge Fehr, and Adam O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 335–359, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany. 3, 14
8. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. 14
9. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 533–556. Springer, 2014. 4
10. Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 108–125, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany. 9, 10, 25
11. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. 14
12. 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, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 535–564, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. 3, 6

13. 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, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 33–65, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 6
14. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. 2
15. Nico Döttling and Sanjam Garg. From selective IBE to full IBE and selective HIBE. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 372–408, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany. 3, 4
16. Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 537–569, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 3, 4, 6
17. Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Daniel Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 3–31, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany. 6
18. Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2019, Part III*, LNCS, pages 3–32, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 2
19. David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 279–295, Paris, France, May 26–28, 2010. Springer, Heidelberg, Germany. 2
20. Sanjam Garg, Romain Gay, and Mohammad Hajiabadi. New techniques for efficient trapdoor functions and applications. In Vincent Rijmen and Yuval Ishai, editors, *EUROCRYPT 2019, Part III*, LNCS, pages 33–63, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany. 2, 5, 7
21. Sanjam Garg and Mohammad Hajiabadi. Trapdoor functions from the computational Diffie-Hellman assumption. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 362–391, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. 2, 6
22. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206, Victoria, BC, Canada, May 17–20, 2008. ACM Press. 9, 11, 27
23. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 503–523, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany. 16
24. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press. Available as Cryptology ePrint Archive Report 2006/309. 14

25. Susan Hohenberger, Venkata Koppula, and Brent Waters. Chosen ciphertext security from injective trapdoor functions. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2020, Part I*, LNCS, pages 836–866, Santa Barbara, CA, USA, August 16–20, 2020. Springer, Heidelberg, Germany. 2, 3
26. Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st ACM STOC*, pages 12–24, Seattle, WA, USA, May 15–17, 1989. ACM Press. 13
27. Eike Kiltz, Daniel Masny, and Krzysztof Pietrzak. Simple chosen-ciphertext security from low-noise LPN. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 1–18, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany. 2
28. Fuyuki Kitagawa, Takahiro Matsuda, and Keisuke Tanaka. CCA security and trapdoor functions via key-dependent-message security. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2019, Part III*, LNCS, pages 33–64, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 11, 12, 15
29. Venkata Koppula and Brent Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2019, Part II*, LNCS, pages 671–700, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 4, 5, 6, 15
30. Yehuda Lindell and Benny Pinkas. A proof of security of Yao's protocol for two-party computation. *Journal of Cryptology*, 22(2):161–188, April 2009. 9, 23, 27
31. Alex Lombardi, Willy Quach, Ron D. Rothblum, Daniel Wichs, and David J. Wu. New constructions of reusable designated-verifier NIZKs. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2019, Part III*, LNCS, pages 670–700, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 3, 12, 15
32. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 187–196, Victoria, BC, Canada, May 17–20, 2008. ACM Press. 2
33. Michael O. Rabin. Digital signatures and public key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology, January 1979. 2
34. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press. 9, 11, 27
35. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978. 2
36. Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany. 3
37. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167, Toronto, Ontario, Canada, October 27–29, 1986. IEEE Computer Society Press. 9, 23, 27