On the Adaptive Security of Free-XOR-based Garbling Schemes in the Plain Model

Anasuya Acharya[1], Karen Azari[2⋆], and Chethan Kamath[3]

[1] Bar-Ilan University, Faculty of Engineering, Israel
acharya@biu.ac.il

[2] University of Vienna, Faculty of Computer Science, Vienna, Austria
karen.azari@univie.ac.at

[3] IIT Bombay, India
ckamath@cse.iitb.ac.in

Abstract. A Garbling Scheme is a fundamental cryptographic primitive, with numerous theoretical and practical applications. Since its inception by Yao (FOCS'82, '86), optimizing the communication and computation complexities of securely garbling circuits has been an area of active research. One such optimization, and perhaps the most fundamental, is the 'Free-XOR' technique (Kolesnikov and Schneider, ICALP'08) which allows XOR gates in a function garbling to not require representation, and therefore communication.

Since then, several works have designed and analysed the security of schemes that adopt the Free-XOR optimisation. In particular: (1) Applebaum (JoC'16) proved that this can be securely instantiated assuming symmetric-key encryption satisfying a notion called RK-KDM security; and (2) Zahur, Rosulek and Evans (Eurocrypt'15) proposed the so-called 'Half Gates' scheme, and proved that it can be instantiated assuming hash functions satisfying a notion called CCR security. Although both schemes have been proven selectively secure, prior work leaves it open to analyze whether they satisfy a stronger security notion – adaptive security – in the plain model.

In this work, we formally show that the selective security of these two schemes *cannot* be lifted to adaptive security under the same assumptions. To establish these barriers, we adopt techniques from the work of Kamath et al (Crypto'21), who proved similar negative results for Yao's garbling. We use that as a starting point and introduce new techniques tailored towards addressing Free-XOR-based schemes.

Keywords: Garbling · Adaptive Security · Free-XOR · Feasibility Lower Bounds

⋆ Part of the work was done while the author was affiliated with ETH Zurich, Switzerland.

1 Introduction

Garbling was introduced by Yao in [Yao86] as a technique towards achieving secure function-evaluation. On a high level, a garbling scheme allows to efficiently encode a circuit $\mathbf{C}$ and an input $\mathbf{x}$ into a *garbled circuit* $\tilde{\mathbf{C}}$ and a *garbled input* $\tilde{\mathbf{x}}$ such that anyone can efficiently derive $\mathbf{C}(\mathbf{x})$ given $\tilde{\mathbf{C}}$ and $\tilde{\mathbf{x}}$. Security requires $\tilde{\mathbf{C}}$ and $\tilde{\mathbf{x}}$ not to leak non-trivial information about $\mathbf{x}$ and $\mathbf{C}$. Garbling is nowadays considered to be a stand-alone cryptographic primitive with theoretical and practical applications, not only in secure function-evaluation but also in other areas like verifiable computation (see [AIK11,App16] and the references therein). In fact, it is explicitly mentioned in the call for multi-party threshold schemes by NIST, listed as a 2nd category primitive [BP23].

Yao's construction and optimisations. The basic construction for Yao's garbling, formalized in [LP09], presents an algorithm that garbles any Boolean circuit $\mathbf{C}$ gate-by-gate using a symmetric-key encryption scheme (SKE). The algorithm works by first sampling, for each wire w in the circuit, n-bit values L_w^0 and L_w^1, called *labels*, that represent the logical 0 and 1 value respectively. Then the circuit is parsed as a set of binary gates and the truth-table of each gate functionality g is 'encrypted'. This is done by generating a ciphertext corresponding to each of the four rows of this truth-table. Such a ciphertext typically is an encryption of an output wire label $\mathsf{L}_C^{g(a,b)}$ using a pair of input wire labels – one from each input wire $(\mathsf{L}_A^a, \mathsf{L}_B^b)$ – as the keys, where the logical value of each label follows from the truth-table row being encoded. This set of four ciphertexts is then randomly permuted, completing the gate garbling. A set of such garbled gates constitute the garbled circuit (GC) $\tilde{\mathbf{C}}$. The garbling algorithm produces this, along with a *decoding map* from output wire labels to their logical value, as output.

In order to evaluate the garbling on a particular input $\mathbf{x}$, it becomes necessary to first refer to the set $\tilde{\mathbf{x}}$ containing the labels corresponding to these input bits. This set contains one label from each input wire. Then the GC $\tilde{\mathbf{C}}$ can be evaluated gate-by-gate, maintaining the invariant that only one label of each wire is ever revealed. This is because, starting with one label each from the input wires, only one of the four ciphertexts in each gate garbling can be correctly decrypted using the unique key-pair available, revealing only one gate output label. There exist different mechanisms for recognizing whether decryption works properly, and for the basic Yao's scheme one would simply try to decrypt all four ciphertexts to find the right one. Finally, when the circuit output labels have been derived, the decoding map can be used to derive the circuit output. The computational complexity of this garbling construction is four encryptions (resp., decryptions) per gate for the party generating $\tilde{\mathbf{C}}$ (resp., the party evaluating $\tilde{\mathbf{C}}$); its communication complexity is (at least) four ciphertexts per gate.

Towards more practical constructions, several optimizations for the above construction have been proposed. We discuss those that will be relevant to our discussion (other optimisations include [BMR90,PSSW09,KMR14,GLNP15, RR21]). First, Naor, Pinkas and Sumner [NPS99] proposed the so-called point-and-permute optimisation, which allows to reduce to cost of evaluating the garbled circuit from four to just a single decryption per gate. With regards to applications that do not require circuit privacy, certain gate functionalities can be garbled more efficiently than others. In [KS08], Kolesnikov and Schneider proposed an optimisation that allows garbling XOR gates for *free*, i.e. the garbling tables of XOR gates are empty. The idea behind this free-XOR optimisation is to sample the labels $(\mathsf{L}_w^0, \mathsf{L}_w^1)$ associated with each wire w with a fixed *global* offset instead of independently uniformly at random. Evaluating an XOR gate in such a garbling involves simply computing a bit-wise XOR of the labels: $\mathsf{L}_C^{a \oplus b} = \mathsf{L}_A^a \oplus \mathsf{L}_B^b$. This results in two distinct labels with the same global offset for the output wire, that are derived from the input labels. Among a long list of garbling schemes that support free-XOR, a scheme used in most practical applications [CWYY23, GKWY20, CRS20, GKW+20, LWN+15, CCPS19] is the Half Gates scheme by Zahur, Rosulek and Evans [ZRE15], which additionally reduces the complexity of garbling AND gates to just two ciphertexts by clever use of (deterministic) hash functions instead of (randomized) encryption.

Security of Yao's garbling and its variants. Informally, the security for a garbling scheme usually requires that the garbled circuit and input encoding be simulatable from the function output only – the garbling leaks no information beyond the output about the circuit and input. However, in several applications (e.g., secure function-evaluation) leakage about the circuit is tolerable, and only *input privacy* suffices. That is, $\tilde{\mathbf{C}}$ and $\tilde{\mathbf{x}}$ are efficiently simulatable given $\mathbf{C}$ and the output $\mathbf{C}(\mathbf{x})$. There are several ways to formalise this requirement, but in this paper we are interested in one particular aspect: selective vs. adaptive security.

Selective security models a setting where the circuit and the input are garbled at the same time and the adversary cannot see the garbled circuit before submitting its input. However, often (e.g., in Yao's construction above) the costliest part of a garbling protocol is garbling of the circuit. Therefore, especially in applications where the circuit to be computed is known publicly and ahead of time, it is often convenient to garble the circuit in an *offline* pre-computation phase, before the input is even chosen. This scenario is not covered by the notion of selective security, but would require stronger *adaptive* security, a notion introduced by Bellare, Hoang, and Rogaway in [BHR12a]. Informally, adaptive security requires that the function garbling $\tilde{\mathbf{C}}$ be simulated independently of the function output and the rest of the garbling can then be simulated using $\mathbf{C}(\mathbf{x})$. This is considered significantly harder to obtain than selective security.

Yao's scheme was proven selectively secure by Lindell and Pinkas [LP09] under the minimal assumption that one-way functions exist. Selective security of the free-XOR optimisation was shown by Applebaum in [App16] based on related-key key-dependent message (RK-KDM) security of the underlying SKE, a new security notion that was introduced in that work. Loosely speaking, RK-KDM security requires ciphertext indistinguishability to hold when *additionally* given access to a "leakage oracle" that allows access to ciphertexts generated using keys and messages *related* to the challenge key. Applebaum constructed such RK-KDM-secure SKE under the learning parity with noise assumption; constructions based on other standard hardness assumptions followed [BDH14].

In order to prove the Half Gates scheme by Zahur, Rosulek and Evans [ZRE15] secure, a novel security notion for hash functions was introduced by Choi et al. [CKKZ12], namely circular correlation robustness (CCR), and can be seen as an analogous notion for hash function (although CCR predates RK-KDM). While RK-KDM secure encryption can be instantiated from various standard computational hardness assumptions [App16, BDH14], for long the only known constructions of CCR secure hash functions are in idealised models such as the random permutation model [GKWY20, GKW+20, CT21]. Very recently, Acharya et al [AAB+25] showed how to securely instantiate (weakly) CCR secure hash functions – and in particular also the Half Gates scheme – in the standard model, but relying on indistinguishability obfuscation.

On the other hand, adaptive security of Yao's garbling is a delicate matter. Although Applebaum et al. [AIKW13] proved impossibility of adaptive security of the original construction, a variant of it (where the output map is sent together with the input garbling) was later proven adaptively secure by Jafargholi and Wichs [JW16] using pebbling-based techniques. However, their result applies only to circuits of *bounded* (logarithmic) depth, and this limitation was later shown to be inherent by Kamath et al. in [KKPW21b]. Not much was known regarding adaptive security of practical garbling constructions. In the standard model (i.e., without resorting to idealised objects), Jafargholi and Oechsner give negative evidence in [JO20] by arguing why the techniques used in [JW16] for Yao's scheme will likely not work in this setting – although this still leaves room for other types of reductions. However, very recently [GYW+23, BHKO23] proved (among other results) that the Half Gates construction is adaptively secure in idealised models (the random oracle/permutation model). Thus we are left with the following open question:

Are practical garbling schemes (i.e. free-XOR, Half Gates) adaptively secure in the standard model?

1.1 Our Results

In this paper, we approach this question for free-XOR-based garbling schemes from a negative perspective. Our first result pertains to Applebaum's construction [App16]. Recall from earlier discussion that it was shown to be selectively secure based on RK-KDM security of the underlying SKE. To be precise, RK-KDM security with respect to *linear* relations – LIN-RK-KDM, in short – suffices. We prove that this construction *cannot* be proved adaptively secure under the same assumption. More formally:

Theorem 1 (informal, see Theorem 3). *Any fully black-box security reduction proving adaptive security of Applebaum's scheme based on LIN-RK-KDM-security of the underlying SKE must incur a loss in security exponential in the security parameter. This is true even when restricting to the circuit class* NC^1.

Note that by a *fully black-box security reduction*[4], we refer to a polynomial-time randomised Turing reduction that has black-box access to the adversary that breaks the construction and the primitive. It can rewind the adversary a polynomial number of times.

We point out that the lower bound in Theorem 1 is much stronger than what [KKPW21b] established for Yao's construction: there, the loss in security grew exponential but only in depth of the circuit (which is optimal given the [JW16] result). Thus the theorem indicates that pebbling-based security reductions [JW16, JSW17, JKK+17, KKP21], which were so useful when it comes to adaptive security of Yao's construction and its variants will not work for Applebaum's construction (confirming the conjecture from [JO20]).

Our second result extends Theorem 1 to the Half Gates scheme [ZRE15]. Recall that Half Gates was shown to be selectively secure based on CCR security of the underlying hash function. We prove that this construction *cannot* be proved adaptively secure under the same assumption. More formally:

Theorem 2 (informal, see Theorem 4). *Any fully black-box security reduction proving adaptive security of the Half Gates scheme based on CCR-security of the underlying hash function must incur a loss in security exponential in the security parameter. This is true even when restricting to the circuit class* NC^1.

This is in contrast with the positive result from [BHKO23, GYW+23], which proved its adaptive security in models where the CCR hash function is modelled as an ideal primitive (see discussion below). We refer to Section 4 for details.

Interpreting our results.

- While we do not find any weakness in the adaptive security of the proposed schemes, our results affirm that the difficulty in showing adaptive security of these schemes based on the current hardness assumptions is inherent (and not due to a lack of understanding). Thus, the random guessing technique (sometimes known as complexity leveraging) is optimal when it comes to the adaptive security of these schemes under the respective assumptions.
- To add to the point above, as in [KKPW21b], we are not establishing a full-blown oracle separation. As pointed out in Footnote 4, we only rule out black-box security reductions for *fixed* constructions. In fact, there exist several garbling schemes that are adaptively secure assuming just one-way functions [JW16, HJO+16, JSW17].
- We emphasise that our negative result is not in contradiction with positive results from [BHKO23, GYW+23]. Their results pertain to idealised settings, where all parties (including the adversary and reduction) have bounded query access to the idealised object. On the other hand, to establish our

[4] This is not to be confused with "fully black-box reduction" of one primitive to another (e.g., adaptive garbling to RK-KDM security) in the [RTV04] sense, that involves a black-box *scheme and* a fully black-box security reduction. We focus on *fixed* black-box constructions, and rule out all black-box *security* reductions for that construction.

result our adversary needs to query the idealised object at unbounded-many points. In fact, for such range of parameters, the result in [GYW$^+$23] is vacuous (their result degrades with the number of queries to the idealised object).

Open questions.

- Although our results presently only serve as limits to provable security of the two schemes, we are optimistic that they can be lifted to concrete counter-examples. This is in the same vein as the work of Hofheinz, Rao and Wichs [HRW16], who constructed an IND-CPA public-key encryption scheme that is *not* secure under the stronger selective-opening attacks (using a strong form of obfuscation). Analogously, it is an interesting open question to construct a concrete RK-KDM secure SKE scheme (resp., CCR-secure hash function) such that Applebaum's scheme (resp., Half Gates scheme) is selectively secure but not adaptively secure.
- Another interesting open question is to prove barriers for other common optimisation techniques. The current approach is tailored to the free-XOR optimisation, and extending it to other optimisations will likely require different techniques.
- Recall that LIN-RK-KDM suffices to show selective security of Applebaum's scheme, but not its adaptive security. An interesting open question is to prove its adaptive security using RK-KDM-secure SKE (i.e., exploiting non-linear leakage functions) or, to continue the theme of our work, extend Theorem 1 to rule out adaptive security even given RK-KDM-secure SKE.

1.2 Technical Overview

In this technical overview, we focus on Theorem 1, demonstrating all the key techniques required for the proof. The proof of Theorem 2 builds on that of Theorem 1, and thus we defer its discussion to the end of this section.

High-level approach. We follow the proof template used in [KKPW21a,KKPW21b] for establishing fine-grained lower bounds on loss in security incurred by *fully black-box security reductions* – henceforth referred simply to as "reduction". They consider a setting where some fixed construction $\Gamma^{(\cdot)}$ that uses an underlying cryptographic primitive (as black box, indicated by $(\cdot)$ in the exponent) is shown secure using a reduction that, given black-box access to an adversary that breaks Γ^E for *any* instantiation E of the primitive, breaks E. In order to prove that the loss in security is exponential, one needs to show that for *every* reduction R there exists (i) an ideal instantiation E of the primitive, and (ii) a possibly-inefficient adversary A that breaks the security of Γ^E, in such a way that loss in security incurred by R is large. The adversarial strategy for A typically involves breaking Γ^E by first breaking the security of E by brute force. The crucial step, which completes the proof, is to then show that R *cannot* exploit A to *itself* break E.

For instance, in [KKPW21b] the construction $\Gamma^{(\cdot)}$ corresponds to Yao's construction of garbled circuits and the underlying primitive was an IND-CPA-secure symmetric key encryption scheme (SKE). Their goal was to show that any reduction R of the adaptive security of Yao's garbling to the security of the SKE would incur a loss in security that is exponential in the depth of the circuit. Towards this, they consider an ideal instantiation E of the SKE and a brute-force adversary as mentioned above. Their main technical idea was to use pebbling lower bounds to establish that the reduction R cannot use A to *itself* break E.

For the case of Theorem 1, E will be a LIN-RK-KDM-secure SKE and $\Gamma^{(\cdot)}$ will be Applebaum's garbling construction [App16]. Our goal is to similarly show that any reduction R of the adaptive security of this garbling scheme to the LIN-RK-KDM-security of SKE would incur loss in security that is exponential in the input length n of the circuit being garbled. As per the template:

- We design an ideal LIN-RK-KDM-secure SKE scheme $E = (\mathsf{Enc}, \mathsf{Dec})$ based on a random injective (expanding) function oracle: i.e., Enc simply evaluates the random oracle on key, message and random coins, and Dec is then defined to be consistent with Enc (which is not necessarily efficient). It can be easily shown that E is LIN-RK-KDM-secure.
- The inefficient adversary A that breaks Γ^E does so by essentially breaking E by brute force (details soon).

However, in order to show that A's strategy cannot be exploited by R we use techniques that significantly depart from the pebbling-based approach in [KKPW21a, KKPW21b], and this constitutes our main technical contribution. We refer the reader to Remark 1 for a discussion on why the pebbling-based approach does not seem to be useful for Theorem 1.

In the rest of the technical overview we proceed as follows. First we recall the relevant security notions for garbling, and describe a simplified version of Applebaum's construction that suffices for explaining our technique. Next we describe our (inefficient) adversarial strategy A, in particular the circuit $\mathbf{C}_d$ it uses. Finally, we explain why A cannot be exploited by R.

Security model and Applebaum's construction. Let us first take a look at the notion of security for garbling slightly more formally. For this technical overview, we restrict ourselves to selective and adaptive *input privacy*, denoted sPRIV-IND and PRIV-IND, respectively. The notion of PRIV-IND is modelled using the following game between a challenger C and an adversary A.

1. *Offline Phase:* A challenges C on a circuit $\mathbf{C} : \{0,1\}^n \to \{0,1\}^m$ and receives a garbled circuit $\tilde{\mathbf{C}}$ in response.
2. *Online Phase:* A selects two inputs $\mathbf{x}_0, \mathbf{x}_1 \in \{0,1\}^n$ such that $\mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1)$, and gives it to the challenger. C returns $\tilde{\mathbf{x}}$, the input encoding corresponding to the garbling, of the input $\mathbf{x}_b$, for a uniformly random bit b. Furthermore, A also returns the decoding function $\mathbf{d}$, which allows to evaluate $\tilde{\mathbf{C}}$ on $\tilde{\mathbf{x}}$ and receive $\mathbf{C}(\mathbf{x}_b)$.

3. A outputs a guess b' and wins if the guess is correct (i.e., $b' = b$).

Note that our definition of the security game PRIV-IND slightly differs from the original definition by Bellare, Hoang and Rogaway [BHR12a]: Namely, we include the decoding function $\mathbf{d}$ in the *online* phase. The reason for this is that Yao's original scheme as well as the schemes considered in this work all succumb to a known lower bound from [AIKW13], which states that in order to guarantee adaptive security, the online complexity must exceed the input *and* the output length. Sending $\mathbf{d}$ in the online phase allows to circumvent this lower bound and prove security at least for certain restricted cases (e.g. for Yao's scheme when applied to circuits of bounded depth [JW16]).

In sPRIV-IND, there is no offline phase: A needs to submit $\mathbf{C}$ and $(\mathbf{x}_0, \mathbf{x}_1)$ at the beginning of the game and is then given $(\tilde{\mathbf{C}}, \tilde{\mathbf{x}}, \mathbf{d})$. Thus the notion is weaker than PRIV-IND.

Next we describe a vanilla version (ignoring so-called "permutation bits", that allow for significant performance improvements) of Applebaum's construction that suffices to convey our techniques. The construction, described below for the XOR-AND basis, proceeds essentially as in Yao's construction described earlier except for the Free-XOR optimisation (for now, we ignore garbled circuit evaluation).

– To garble a circuit $\mathbf{C} : \{0,1\}^n \to \{0,1\}^m$, first a *global offset* Δ is sampled uniformly at random from the space of labels.
  1. For each wire w in $\mathbf{C}$ that is either a circuit input wire, or the output wire of some AND gate, sample the 0-label k_w^0 uniformly at random and set the 1-label as $k_w^1 := k_w^0 \oplus \Delta$.
  2. For every XOR gate (in topological order) with input wires (A, B) and output wire C, set $k_C^0 := k_A^0 \oplus k_B^0$ and $k_C^1 := k_C^0 \oplus \Delta$. The garbling table of XOR gates is empty.
  3. The garbling table of an AND gate g with input wires A, B and output wire C consists of four (pairs of) ciphertexts, each encoding one row of the truth-table of the binary AND gate. For each row, for $(a, b) \in \{0,1\}^2$, the first ciphertext is an encryption of a randomly-sampled "mask" R under k_A^a; and the second ciphertext is an encryption of the masked target key $R \oplus k_C^{a \wedge b}$ under k_B^b. Thus, the garbling table for AND has the following form:

$$\begin{aligned}
\pi[00] &: (\mathsf{Enc}_{k_A^0}(R_{00}), \mathsf{Enc}_{k_B^0}(R_{00} \oplus k_C^0)) \\
\pi[01] &: (\mathsf{Enc}_{k_A^0}(R_{01}), \mathsf{Enc}_{k_B^1}(R_{01} \oplus k_C^0)) \\
\pi[10] &: (\mathsf{Enc}_{k_A^1}(R_{10}), \mathsf{Enc}_{k_B^0}(R_{10} \oplus k_C^0)) \\
\pi[11] &: (\mathsf{Enc}_{k_A^1}(R_{11}), \mathsf{Enc}_{k_B^1}(R_{11} \oplus k_C^1))
\end{aligned} \tag{1}$$

  The garbled circuit $\tilde{\mathbf{C}}$ consists of the garbling tables π (with the four pairs of ciphertexts $\pi[ab]$ in random order) for each AND gate in the circuit $\mathbf{C}$.
– The input encoding $\tilde{\mathbf{x}}$ for circuit input $\mathbf{x} \in \{0,1\}^n$ is a set of labels that contains $\mathbf{x}[i]$-label of (input) wire i for each $i \in [n]$.
– The output decoding function $\mathbf{d}$ simply maps the labels of the circuit output wires to their actual semantic values.

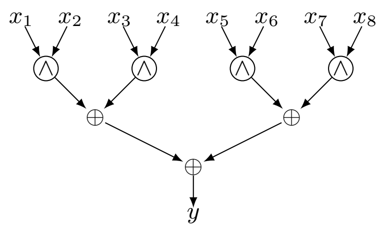

Fig. 1. The circuit $\mathbf{C}_3$ for $n = 8 = 2^3$.

Our adversarial strategy A. Let us first describe the circuit that the adversary A uses in its strategy for the adaptive security game. Letting n be the required circuit input size, let $d = \lceil \log_2 n \rceil \in \mathbb{N}$ parameterize this circuit $\mathbf{C}_d$. We set the topology of this circuit as a binary tree of depth d where the first layer consists of AND gates (thus all the input wires are connected to AND), while the rest of the gates are all XOR gates (see Figure 1).[5] The rationale behind this design will become clear soon.

For the technical overview, let us restrict to *non-rewinding* reductions R that invoke the adversary only once on security parameters that are powers of two, and cannot control the adversary's randomness – in the main body, we extend the ideas to arbitrary reductions using a q-wise independent hash function, for an appropriately-chosen q. For security parameter κ, setting $n = \kappa$, the adversary A when invoked on an input 1^n, where $n = 2^d$ for $d \in \mathbb{N}$, does the following:

1. Send $\mathbf{C}_d$ to the challenger and receive the garbled circuit $\tilde{\mathbf{C}}$ in return. Note that this comprises of only the garbled table representations of the first layer, since the XOR gates are garbled "for free".
2. Sample random $\mathbf{x}_1 \in \{0,1\}^n$ and compute a random "colliding" input $\mathbf{x}_0$ (i.e. s.t. $\mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1)$), and receive in return an input encoding $\tilde{\mathbf{x}}$ for one of these inputs.

To determine its guess b' for the input encoded, A proceeds in two steps:

– First it carries out a sequence of (inefficient) checks of "well-formedness" of $\tilde{\mathbf{C}}$ to determine whether the garbled AND gate truth-tables in $\tilde{\mathbf{C}}$ have the structure of honestly-generated garbled gates (as in Eq. (1)). To this end, it brute-forces the encryption oracle of E to extract all keys and messages associated with the ciphertexts in $\tilde{\mathbf{C}}$ (since Enc is a random expanding function, with high probability every ciphertext is associated with a unique key/message pair).
– Second it checks if $(\tilde{\mathbf{C}}, \tilde{\mathbf{x}})$ is consistent with $\mathbf{x}_1$. This requires A to exploit the asymmetry in (honest) garbling/gate truth-table of AND gates, which allows it to construct a logical mapping from the secret keys it extracted earlier to their logical bit value (hence the need for the first layer of AND gates in $\mathbf{C}_d$).

[5] The choice of gates for the bottom layers is arbitrary: the analysis only relies on the first layer being all AND gates: see Footnote 7.

If all the checks pass, A outputs 1; otherwise it outputs 0. It is easy to see that A is a valid adversary against PRIV-IND security of Γ^E: in particular, in the games corresponding to the challenger C returning an honestly generated garbling of the circuit and input $\mathbf{x}_0$ resp. $\mathbf{x}_1$, it outputs 0 and 1, respectively. We refer the readers to Section 3.2 for a formal definition of A, and Lemma 1 for a proof of why it is a valid PRIV-IND adversary. Next we argue why A is useless to R when it comes to solving its own LIN-RK-KDM challenge.

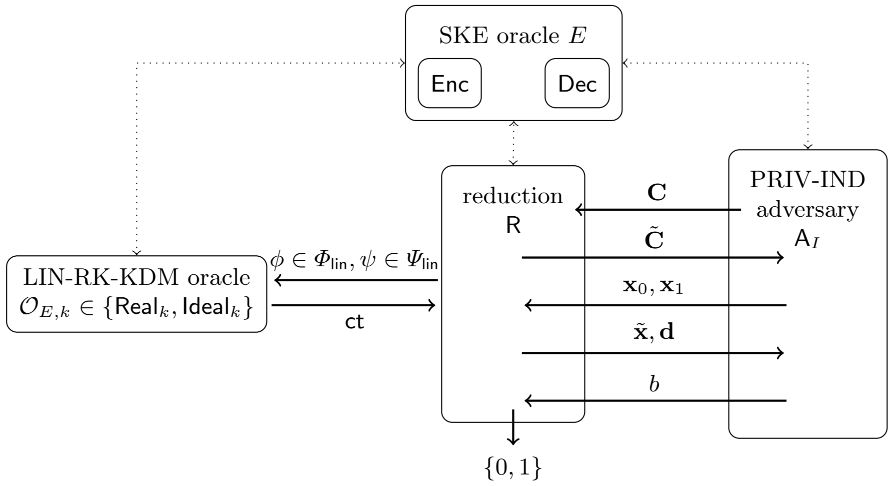
SKE oracle E
Enc
Dec
reduction R
PRIV-IND adversary A_I
LIN-RK-KDM oracle $\mathcal{O}_{E,k} \in \{\mathsf{Real}_k, \mathsf{Ideal}_k\}$
$\phi \in \Phi_{\mathsf{lin}}, \psi \in \Psi_{\mathsf{lin}}$
ct
$\mathbf{C}$
$\tilde{\mathbf{C}}$
$\mathbf{x}_0, \mathbf{x}_1$
$\tilde{\mathbf{x}}, \mathbf{d}$
b
$\{0,1\}$

Fig. 2. Structure of the Reduction

Why can R *not exploit A?* Recall that the reduction R as in Figure 2 aims to break the LIN-RK-KDM security of the encryption scheme E by leveraging A's ability to break PRIV-IND security of Γ^E. The LIN-RK-KDM security notion roughly requires distinguishing ciphertexts under E, when additionally given access to a "leakage oracle". More formally, the reduction R is given access to (i) an "encryption oracle" $\mathcal{O}^{\mathsf{Enc}}$, which on inputs of the form (k, m) outputs $\mathsf{Enc}(k, m)$, and (ii) a leakage oracle $\mathcal{O}^{\mathsf{LIN}}_{b^*}$ for challenge key k^*, that on input $(\Delta_\phi, b, \Delta_\psi)$ outputs $\mathsf{ct} \leftarrow \mathsf{Enc}_{k^* \oplus \Delta_\phi}(b \cdot k^* \oplus \Delta_\psi)$ when $b^* = 0$ or $\mathsf{ct} \leftarrow \mathsf{Enc}_{k^* \oplus \Delta_\phi}(0^n)$ when $b^* = 1$. Its goal is to guess whether the ciphertexts are real ($b^* = 0$) or fake ($b^* = 1$).

Intuitively, for R to solve its own RK-KDM challenge using A, it has to embed (at least one of) the RK-KDM challenge ciphertexts within the AND-gate garbling of the garbled circuit that it supplies to A. However, since E is an ideal SKE, all ciphertexts in $\tilde{\mathbf{C}}$ must be generated either through the $\mathcal{O}^{\mathsf{Enc}}$ oracle or by the LIN-RK-KDM oracle $\mathcal{O}^{\mathsf{LIN}}_{b^*}$ – this is guaranteed as the image of Enc is a sparse random subset of its co-domain. Thanks to the design of $\mathbf{C}_d$, it can be shown that given $\tilde{\mathbf{C}}$ and the set of queries to $\mathcal{O}^{\mathsf{Enc}}$ and $\mathcal{O}^{\mathsf{LIN}}_{b^*}$ made by R, for all but at most one input, viz. $\mathbf{x}_1$, and for all input garblings $\tilde{\mathbf{x}}$, one of the following must hold:

1. The output of A on transcript $(\mathbf{C}_d, \tilde{\mathbf{C}}, (\mathbf{x}_0, \mathbf{x}_1), (\tilde{\mathbf{x}}, \mathbf{d}))$ is 0. This happens if either $\tilde{\mathbf{C}}$ was not a well-formed garbling, or $\tilde{\mathbf{x}}$ was an encoding of input $\mathbf{x}_0$.
2. When the output of A is 1, we have that the transcript contains a well-formed garbling $\tilde{\mathbf{C}}$ of $\mathbf{C}_d$ and that $\tilde{\mathbf{x}}$ was an input encoding of $\mathbf{x}_1$. In this case, the reduction R, given $\tilde{\mathbf{x}}$ and the list of queries it made to oracles $\mathcal{O}^{\mathsf{Enc}}$ and $\mathcal{O}_{b^*}^{\mathsf{LIN}}$ before sending $\tilde{\mathbf{C}}$, can efficiently extract all involved keys by extracting the global offset Δ.

Thus for R to exploit A, it must have essentially guessed $\mathbf{x}_1$. Since we define A to sample $\mathbf{x}_1 \in \{0,1\}^n$ uniformly at random, the probability of R exploiting A is exponentially-small.

To turn this intuition into a formal argument, we first show that given access to R's queries to the $\mathcal{O}^{\mathsf{Enc}}$ and $\mathcal{O}_{b^*}^{\mathsf{LIN}}$ oracles, A can be "simulated" using an *efficient* algorithm $\hat{\mathsf{A}}$ (with overwhelming probability). As noted above, since E is an ideal SKE, R must have obtained all ciphertexts in $\tilde{\mathbf{C}}$ via queries to $\mathcal{O}^{\mathsf{Enc}}$ and $\mathcal{O}_{b^*}^{\mathsf{LIN}}$ oracles. We show that these queries contain sufficient information to carry out "well-formedness" and consistency checks made by A in an efficient manner (see Section 3.3 for exactly how). Then we show that R (which, recall, is efficient) can distinguish A from $\hat{\mathsf{A}}$ with probability at most 2^{-n}. Formally, we first show in Lemma 2 (Section 3) that R distinguishes 1) the world where it is given oracle access to A and the real LIN-RK-KDM oracle; and 2) the world where it is given oracle access to $\hat{\mathsf{A}}$ and the real LIN-RK-KDM oracle, with only an exponentially-small probability. Then, in Lemma 3 we show that a similar claim holds when the real LIN-RK-KDM oracle is replaced by the ideal LIN-RK-KDM oracle. Together, Lemmas 2 and 3 imply that R's advantage in breaking LIN-RK-KDM security when given access to A is exponentially-close to its advantage when given access to $\hat{\mathsf{A}}$. However, since $\hat{\mathsf{A}}$ is efficient and can be simulated using R's queries, this implies that if R is a valid reduction then it, by itself, can break E's LIN-RK-KDM security, without A's help – yielding a contradiction. We note that the overall technique used above is reminiscent, on a high level, of meta-reductions [BV96], especially the proof in [GW11].

Extending to Theorem 2: how to deal with shrinking functions? Before explaining how Theorem 1 can be extended to show Theorem 2, it is instructive to see why similar techniques should even work. On a high level, both Half Gates and Applebaum's scheme can be regarded to be instantiations of the same template to garbling, but using different primitives.

Thus, to extend our results to the Half Gates scheme, we again model the CCR hash function H underlying the scheme as an idealised function. The major difference now is that — unlike LIN-RK-KDM-secure encryption — the hash function H is compressing. Furthermore, the Half Gates scheme additionally employs a row reduction technique which reduces the number of ciphertexts in the garbling table of AND to just two – the structure of the resulting garbling table is thus quite different. Taken together, we need to deal with the extraction of the secret global offset Δ with a bit more care (e.g., now we cannot rely on sparsity of the range, and collisions are guaranteed). However, selecting H as a random function such that $\mathsf{H}(\cdot, i) : \{0,1\}^\kappa \to \{0,1\}^\kappa$ is a random permutation for every $i \in \mathbb{N}$ will be convenient and sufficient to exploit most of the ideas we used in Theorem 1. We refer the readers to Section 4 for more details.

Remark 1 (Difference from [KKPW21b, KKPW21a]). A crucial difference to previous lower bounds [KKPW21b,KKPW21a] is that all schemes considered in these works were built on many *independent* instances of another primitive (e.g., SKE or public-key encryption), thereby allowing clever embedding of challenges within the scheme. Towards proving a lower bound, they basically proved that any fully black-box reduction must embed many such challenges in the scheme simultaneously, and with high probability will fail to embed them in a way that is consistent with the adaptively chosen input. In contrast, in the practical garbling schemes considered in this work, the only secret is the global offset Δ, hence, all secrecy of the scheme must rely on Δ. Our approach here is to show that every fully black-box reduction either fails to simulate a garbling that is consistent with the adaptively chosen input, or allows to extract Δ, in which case there is no secrecy left to relate the security of the scheme to.
This global, extraction-based approach allows us to prove an inherent security loss exponential in the *input size*, opposed to the so-called pebbling complexity or *depth* of the circuit, as in [KKPW21b]. This proves that guessing the input (aka complexity leveraging) is essentially the best proof strategy among all fully black-box reductions relating the security of Applebaum's garbling scheme (resp. the Half gates scheme) to LIN-RK-KDM security of the underlying encryption scheme (resp. CCR security of the employed hash function). In particular, our results confirm the conjecture of [JO20] that more sophisticated guessing techniques (e.g. pebbling techniques) as used in [JW16, JKK$^+$17] will not be useful in proving practical garbling schemes secure.

1.3 More Related Work

Achieving adaptive privacy for garbling schemes has been the subject of a long line of research – starting from its introduction in [BHR12a]. Recall that this property requires simulation of a garbled circuit to be possible even when the circuit is chosen first and the input is later chosen after the adversary receives its garbling. [BHR12a] shows that adaptive garbling is trivially achievable in the programmable Random Oracle Model (ROM) – providing constructions for the same. However, for garbling schemes in the plain model, proving adaptive privacy is highly non-trivial as existing methods for simulating the garbling for selective privacy rely on the circuit output being available to the simulator. So these techniques cannot extend to the adaptive setting where the garbling needs to be simulated before the circuit input, and hence output, becomes known.

Applebaum et al. in [AIKW13] present a lower-bound on the achievable online complexity for adaptive garbling schemes, showing that it has to exceed the sizes of the circuit input and output. Following this, a line of works investigate the achievable extent of adaptive privacy for the traditional Yao's garbling scheme [Yao86]. Notably, [JW16] proves that Yao's garbling is adaptively secure when its use is restricted to circuits in NC^1. Much later, this is complemented with [KKPW21b] that presents a feasibility lower-bound for the circuits complexity for which Yao's garbling is adaptively secure. Informally, they show that the security loss for adaptive privacy in Yao's scheme is exponential in the circuit depth. Recently, Brzuska et al [BBK+23] proposed a weaker notion of simulatability with the goal of bypassing the [AIKW13] impossibility (while still be useful enough for some applications).

Given these limitations in proving adaptive privacy for traditional schemes, a different line of works focus on constructing new garbling schemes that are tailored to be adaptively secure. Among these [HJO+16] presents a garbling scheme constructed using a new primitive – somewhere equivocal encryption schemes – which they show can be constructed from one-way functions. This serves as a feasibility result for adaptive garbling in the plain model yielding a size-inefficient garbled circuit. [JSW17] builds on this to present a garbling scheme using the same building-blocks that is secure with respect to the indistinguishability definition of adaptive privacy. In contrast [GS18] constructs an adaptively secure garbling scheme from standard hardness assumptions like the Computational Diffie Hellman and Learning With Errors problem.

While on the one hand new communication-inefficient constructions for adaptively secure garbling were constructed, [JO20] showed that many existing selectively secure garbling schemes that support practical optimisations for garbling size (making them more communication efficient than Yao's scheme) cannot be proven adaptively private using pebbling techniques. They complement this negative result with a construction of an efficient adaptively secure garbling scheme based on PRFs. Their scheme garbles XOR gates with two ciphertexts and AND gates with three ciphertexts per gate respectively.

Most recently, Barnum et al. present a framework for proving adaptive security for garbling schemes in the non-programmable Random Oracle Model (npROM) in [BHKO23]. Their framework applies to certain garbling schemes when their underlying cryptographic object is replaced with calls to a random oracle. They demonstrate this proof in the simulation paradigm in the presence of PPT adversaries where, notably, the simulator does not program the random oracle. However, their proof does leverage programmability in the hybrid experiments within the proof of security. In the non-programmable Random Permutation Model (npRPM) [GYW+23] present a proof for adaptive security of the garbling schemes in [ZRE15] and [RR21] in the simulation paradigm for PPT adversaries. Their proof techniques are specific to these schemes and rely on rigorous probability analysis for proving the indistinguishability of their hybrid experiments.

2 Preliminaries

General Notation. We say that a function $f : \mathbb{N} \to \mathbb{R}$ is *negligible* if for all constants $c > 0$, there exists $N \in \mathbb{N}$ such that for all $n > N$, $f(n) < n^{-c}$. We say f is exponentially-slow growing if $f(n) = 2^{-\Omega(n)}$. We sometimes denote negligible functions by $\mathsf{neg}(\cdot)$. We denote by κ the computational security parameter. Let bold-face letters represent vectors, e.g. $\mathbf{v}$, where $\mathbf{v}[i]$ refers to the element in the i^{th} index of the vector $\mathbf{v} \in \Sigma^n$ over alphabet Σ, and $|\mathbf{v}| := n$ refers to the number of elements in $\mathbf{v}$. We sometimes associate a vector $\mathbf{v} \in \Sigma^n$ with the set $\{\mathbf{v}[i]\}_{i\in[n]} \subseteq \Sigma$ of elements in $\mathbf{v}$. For an integer $n \in \mathbb{N}$ let $[n] := \{1, 2, \ldots, n\}$. We will denote the truth value of a formula by double brackets, e.g. $[\![x = y]\!] = 1$ if and only of $x = y$. For an algorithm A, we write $\mathsf{A}^{\mathcal{O}}$ to denote that A has oracle access to some oracle $\mathcal{O}$. We write $x \leftarrow \mathsf{A}$ to denote that A outputs x when run on uniformly random coins, and $x \leftarrow \mathcal{X}$ to denote sampling uniformly at random from a set $\mathcal{X}$.

Circuit Notation. We represent a Boolean circuit $\mathbf{C} : \{0,1\}^n \to \{0,1\}^m$ as a tuple of the form $\mathbf{C} = (n, m, q, \{A, B, C, f_g\}_{g\in[q]})$ with $f_g \in \{\mathsf{AND}, \mathsf{XOR}\}$. Here q represents the number of gates, n and m the input and output lengths, respectively. For each (binary) gate indexed $g \in [q]$, A and B represent the indices of its left and right input wires respectively, while $C = n + g$ represents the output wire index. f_g represents the functionality of the gate. Note that, in particular, the representation of $\mathbf{C}$ exceeds the number of gates.

For our proofs we require the following security definition for hash functions.

Definition 1 (q-wise independent hash function family [CW79]). *For $q \in \mathbb{N}$, a keyed function $F : \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a q-wise independent hash function if for all $x_1 \neq \cdots \neq x_q \in \mathcal{X}$ and all $y_1, \cdots, y_q \in \mathcal{Y}$,*

$$\Pr_{I\leftarrow\mathcal{I}}[F(I, x_1) = y_1, \cdots, F(I, x_q) = y_q] = 1/|\mathcal{Y}|^q. \tag{2}$$

[WC81] described a construction of q-wise independent hash functions based on polynomial interpolation.

2.1 Garbling Schemes

The definition of garbling as a standalone cryptographic primitive was due to Bellare et al. [BHR12b].

Definition 2 (Garbling Scheme [BHR12b]). *A garbling scheme is a tuple $\Gamma = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathsf{De})$ containing four polynomial-time algorithms, where Gb is randomised and $\mathsf{En}, \mathsf{Ev}, \mathsf{De}$ are deterministic:*

– $(\tilde{\mathbf{C}}, \mathbf{e}, \mathbf{d}) \leftarrow \mathsf{Gb}(1^\kappa, \mathbf{C})$*: on input a security parameter (in unary) and a circuit $\mathbf{C}$, returns a garbling $\tilde{\mathbf{C}}$, input encoding function $\mathbf{e}$, and output decoding function $\mathbf{d}$.*

– $\tilde{\mathbf{x}} = \mathsf{En}(\mathbf{e}, \mathbf{x})$*: returns the encoding* $\tilde{\mathbf{x}}$ *for function input* $\mathbf{x}$*.*
– $\mathbf{y} = \mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}})$*: returns the output labels* $\mathbf{y}$ *by evaluating* $\tilde{\mathbf{C}}$ *on* $\tilde{\mathbf{x}}$*.*
– $\{\perp, \mathbf{y}\} = \mathsf{De}(\mathbf{y}, \mathbf{d})$*: returns either the failure symbol* $\perp$ *or a value* $\mathbf{y} = f(\mathbf{x})$*.*

These algorithms must satisfy the following properties:

– ***Correctness****: For every circuit* $\mathbf{C} : \{0,1\}^n \to \{0,1\}^m$ *and input* $\mathbf{x} \in \{0,1\}^n$*,*

$$\Pr[\mathbf{y} = \mathbf{C}(\mathbf{x}) : (\tilde{\mathbf{C}}, \mathbf{e}, \mathbf{d}) \leftarrow \mathsf{Gb}(\mathbf{C}),\ \tilde{\mathbf{x}} = \mathsf{En}(\mathbf{e}, \mathbf{x}),\ \mathbf{y} = \mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}}),\ \mathbf{y} = \mathsf{De}(\mathbf{d}, \mathbf{y})] = 1$$

– ***Non-degeneracy****: For any pair* $\mathbf{C}_0, \mathbf{C}_1 : \{0,1\}^n \to \{0,1\}^m$ *with* q *gates,*

$$\{\mathbf{e}_0, \mathbf{d}_0\}_{(\tilde{\mathbf{C}}_0, \mathbf{e}_0, \mathbf{d}_0) \leftarrow \mathsf{Gb}(\mathbf{C}_0)} \equiv \{\mathbf{e}_1, \mathbf{d}_1\}_{(\tilde{\mathbf{C}}_1, \mathbf{e}_1, \mathbf{d}_1) \leftarrow \mathsf{Gb}(\mathbf{C}_1)}$$

Garbling schemes are additionally defined to have a *privacy* property and for all traditional schemes, this is the selective privacy property as in Definition 3.

Definition 3 (Selective Privacy by Indistinguishability (sPRIV-IND)). *A garbling scheme* $\Gamma = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathsf{De})$ *(Definition 2) is sPRIV-IND-secure if for every two-stage PPT adversary* $\mathsf{A} = (\mathsf{A}_0, \mathsf{A}_1)$ *the following value is negligible:*

$$\Bigg| \Pr_{\substack{(\mathbf{C}, \mathbf{x}_0, \mathbf{x}_1, \sigma) \leftarrow \mathsf{A}_0(1^\kappa)\ ;\ \mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1) \\ (\tilde{\mathbf{C}}, \mathbf{e}, \mathbf{d}) \leftarrow \Gamma.\mathsf{Gb}(1^\kappa, \mathbf{C})}} [\mathsf{A}_1(\sigma, (\tilde{\mathbf{C}}, \Gamma.\mathsf{En}(\mathbf{e}, \mathbf{x}_0), \mathbf{d})) = 1] - \Pr_{\substack{(\mathbf{C}, \mathbf{x}_0, \mathbf{x}_1, \sigma) \leftarrow \mathsf{A}_0(1^\kappa)\ ;\ \mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1) \\ (\tilde{\mathbf{C}}, \mathbf{e}, \mathbf{d}) \leftarrow \Gamma.\mathsf{Gb}(1^\kappa, \mathbf{C})}} [\mathsf{A}_1(\sigma, (\tilde{\mathbf{C}}, \Gamma.\mathsf{En}(\mathbf{e}, \mathbf{x}_1), \mathbf{d})) = 1] \Bigg|$$

A stronger privacy definition than the above is the notion of adaptive privacy as in Definition 4. This differs from the selective privacy definition in that the adversary first selects a circuit in an offline phase and receives a garbled circuit. It can then select inputs based on this in the online phase. In the definition of adaptive PRIV-IND we depart from the standard definition by outputting the decoding function $\mathbf{d}$ in the online phase, so to avoid the lower bound of Applebaum et al. [AIKW13].

Definition 4 (Adaptive Privacy by Indistinguishability (PRIV-IND)). *Let* $n(\cdot)$ *and* $m(\cdot)$ *be polynomial functions. Let* $\Gamma = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathsf{De})$ *be a garbling scheme (Definition 2) and* κ *a computational security parameter. Consider the following security game between a PPT adversary* A *and a challenger* C *for adaptive privacy by indistinguishability:*

– ***Offline Phase:***
  - A *chooses a circuit* $\mathbf{C} : \{0,1\}^{n(\kappa)} \to \{0,1\}^{m(\kappa)}$ *and gives this to* C
  - C *computes* $(\tilde{\mathbf{C}}, \mathbf{e}, \mathbf{d}) \leftarrow \Gamma.\mathsf{Gb}(1^\kappa, \mathbf{C})$ *and gives* $\tilde{\mathbf{C}}$ *to* A

– ***Online Phase:***
  - A *chooses* $\mathbf{x}_0, \mathbf{x}_1 \in \{0,1\}^{n(\kappa)}$ *s.t.* $\mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1)$ *and gives* $(\mathbf{x}_0, \mathbf{x}_1)$ *to* C
  - C *samples a bit* $b \leftarrow \{0,1\}$ *and computes* $\tilde{\mathbf{x}} = \Gamma.\mathsf{En}(\mathbf{e}, \mathbf{x}_b)$
  - C *gives* $(\tilde{\mathbf{x}}, \mathbf{d})$ *to* A

- A *outputs a bit* b'.

Γ is PRIV-IND-secure if the advantage of every PPT adversary A *is*

$$\mathit{Adv}_{\mathsf{A}}^{PRIV\text{-}IND}(\kappa) := \left| \Pr[b' = b \mid b' \leftarrow \langle \mathsf{A}(1^\kappa), \mathsf{C}(1^\kappa)\rangle] - \frac{1}{2} \right| < \mathsf{neg}(\kappa)$$

for some negligible function neg.

2.2 LIN-RK-KDM based Garbling Scheme of [App16]

We recall the definition of an RK-KDM secure encryption scheme restricted to settings that suffice for this paper. To be specific, we restrict to fixed-length keys and messages, and the class of linear relations on them. Then we describe the construction of a garbling scheme using this primitive that supports free-XOR, as given in [App16].

Definition 5 (LIN-RK-KDM Secure Encryption Scheme). *Let* $\mathsf{E} = (\mathsf{Enc}, \mathsf{Dec})$ *be a symmetric key encryption scheme over message space and key space* $\mathcal{M} = \mathcal{K} = \{0,1\}^\kappa$ *satisfying,*

– ***Correctness:*** $\forall$ *messages* $m \in \mathcal{M}$ *and all keys* $k \in \mathcal{K}$,

$$\Pr[\mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m] = 1.$$

– ***Length Regularity:*** $\forall m_0, m_1 \in \mathcal{M}$, $k \in \mathcal{K}$, $c_0 \leftarrow \mathsf{Enc}(k, m_0)$, $c_1 \leftarrow \mathsf{Enc}(k, m_1)$ *it holds that* $|c_0| = |c_1|$.

Let the associated family of key-derivation functions Φ_{lin} *and key-dependent-message functions* Ψ_{lin} *determining the legal relations between the key-related keys, and the key-related messages be as follows:*

$$\Phi_{\mathsf{lin}} := \left\{ \phi_\Delta : \mathcal{K} \to \mathcal{K} \mid \phi_\Delta(k) = k \oplus \Delta \right\}_{\Delta \in \mathcal{K}}$$

$$\Psi_{\mathsf{lin}} := \left\{ \psi_{b,\Delta} : \mathcal{K} \to \mathcal{M} \mid \psi_{b,\Delta}(k) = b \cdot k \oplus \Delta \right\}_{b \in \{0,1\},\, \Delta \in \mathcal{K}}$$

Let the oracles Real_k *and* Ideal_k, *parameterized by* $k \in \mathcal{K}$ *be of the form:*

$$\mathsf{Enc}(\phi(k), \psi(k)) \leftarrow \mathsf{Real}_k(\phi \in \Phi_{\mathsf{lin}}, \psi \in \Psi_{\mathsf{lin}})$$

$$\mathsf{Enc}(\phi(k), 0^{|\psi(k)|}) \leftarrow \mathsf{Ideal}_k(\phi \in \Phi_{\mathsf{lin}}, \psi \in \Psi_{\mathsf{lin}})$$

E *is semantically secure under Related Key and Key Dependent Message Attacks for Linear Relations (LIN-RK-KDM-secure, for short) if for* $k \leftarrow \mathcal{K}$, *for any PPT adversary* A, *its advantage in distinguishing these two oracles, i.e.*

$$\mathit{Adv}_{\mathsf{A}}^{\mathit{LIN}}(\kappa) := \frac{1}{2} \cdot \left| \Pr\left[\mathsf{A}^{\mathsf{Real}_k}(1^\kappa) = 1\right] - \Pr\left[\mathsf{A}^{\mathsf{Ideal}_k}(1^\kappa) = 1\right] \right|,$$

is negligible in κ.

Based on a LIN-RK-KDM-secure encryption scheme for key and message relations Φ_{lin} and Ψ_{lin}, in [App16], Applebaum constructs a selectively secure (i.e. sPRIV-IND secure) garbling scheme for circuits, that supports free-XOR. Algorithms 1 and 2 detail this scheme. In this paper, we show that LIN-RK-KDM security of the encryption scheme cannot be used in a black-box way to prove *adaptive* security (i.e. PRIV-IND security) of Applebaum's garbling scheme.

Algorithm 1 Garbling Scheme Γ^{E} for circuits with XOR and AND gates

```
1: procedure $\mathsf{Gb}(1^\kappa, \mathbf{C})$
2:     initialize $\tilde{\mathbf{C}} = []$, $\mathbf{e} = []$ and $\mathbf{d} = []$
3:     sample $\Delta \leftarrow \mathcal{K}$ s.t. $\mathsf{lsb}(\Delta) = 1$
4:     for every $i \in [n]$ do
5:         sample $\mathsf{L}_i^0 \leftarrow \mathcal{K}$ and set $\mathsf{L}_i^1 = \mathsf{L}_i^0 \oplus \Delta$
6:         set $\mathbf{e}[i] = \mathsf{L}_i^0$
7:     end for
8:     for each $g \in [q]$ do
9:         parse $g$th gate as $(A, B, C, f_g)$
10:        if $f_g == \mathsf{XOR}$ then
11:            set $\mathsf{L}_C^0 = \mathsf{L}_A^0 \oplus \mathsf{L}_B^0$ and $\mathsf{L}_C^1 = \mathsf{L}_C^0 \oplus \Delta$
12:        else
13:            let $c_A = \mathsf{lsb}(\mathsf{L}_A^0)$, $c_B = \mathsf{lsb}(\mathsf{L}_B^0)$, sample $\mathsf{L}_C^0 \leftarrow \mathcal{K}$ and set $\mathsf{L}_C^1 = \mathsf{L}_C^0 \oplus \Delta$
14:            compute and set $\tilde{\mathbf{C}}[g] = \mathbf{G}_g$ where,
               $\mathbf{G}_g[c_A, c_B] = (\mathsf{Enc}(\mathsf{L}_A^0, R_1); \mathsf{Enc}(\mathsf{L}_B^0, R_1 \oplus \mathsf{L}_C^0))$        $R_1 \leftarrow \mathcal{K}$
               $\mathbf{G}_g[c_A, \neg c_B] = (\mathsf{Enc}(\mathsf{L}_A^0, R_2); \mathsf{Enc}(\mathsf{L}_B^1, R_2 \oplus \mathsf{L}_C^0))$       $R_2 \leftarrow \mathcal{K}$
               $\mathbf{G}_g[\neg c_A, c_B] = (\mathsf{Enc}(\mathsf{L}_A^1, R_3); \mathsf{Enc}(\mathsf{L}_B^0, R_3 \oplus \mathsf{L}_C^0))$       $R_3 \leftarrow \mathcal{K}$
               $\mathbf{G}_g[\neg c_A, \neg c_B] = (\mathsf{Enc}(\mathsf{L}_A^1, R_4); \mathsf{Enc}(\mathsf{L}_B^1, R_4 \oplus \mathsf{L}_C^1))$      $R_4 \leftarrow \mathcal{K}$
15:        end if
16:    end for
17:    for each $j \in [m]$ do
18:        set $\mathbf{d}[j] = \mathsf{lsb}(\mathsf{L}_j^0)$
19:    end for
20:    return $(\tilde{\mathbf{C}}, (\Delta, \mathbf{e}), \mathbf{d})$
21: end procedure
22:
23: procedure $\mathsf{En}((\Delta, \mathbf{e}), \mathbf{x})$
24:    initialize $\tilde{\mathbf{x}} = []$
25:    for every $i \in [n]$ do
26:        set $\tilde{\mathbf{x}}[i] = \mathbf{e}[i] \oplus \mathbf{x}[i]\Delta$
27:    end for
28:    return $\tilde{\mathbf{x}}$
29: end procedure
```

Algorithm 2 Algorithms to Evaluate the Garbling in Γ^{E}

```
procedure $\mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}})$
    initialize $\mathbf{y} = []$
    for each $g \in [q]$ do
        $\mathsf{L}_A, \mathsf{L}_B \leftarrow$ active labels associated with the input wires of $g$th gate
        if $f_g == \mathsf{XOR}$ then
            $\mathsf{L}_C = \mathsf{L}_A \oplus \mathsf{L}_B$
        else
            for $s_A = \mathsf{lsb}(\mathsf{L}_A)$, $s_B = \mathsf{lsb}(\mathsf{L}_B)$, let $(c_0, c_1) = \mathbf{G}_g[s_A, s_B]$
            compute $\mathsf{L}_C = \mathsf{Dec}(\mathsf{L}_A, c_0) \oplus \mathsf{Dec}(\mathsf{L}_B, c_1)$
        end if
        if $C$ is a circuit output wire then
            $\mathbf{y}[C] = \mathsf{L}_C$
        end if
    end for
    return $\mathbf{y}$
end procedure

procedure $\mathsf{De}(\mathbf{y}, \mathbf{d})$
    initialize $\mathbf{y}' = []$
    for $j \in [m]$ do
        $\mathbf{y}'[j] = \mathbf{d}[j] \oplus \mathsf{lsb}(\mathbf{y}[j])$
    end for
    return $\mathbf{y}'$
end procedure
```

2.3 CCR Hash based Garbling Scheme of [ZRE15]

We recall the definition of circular correlation robust (CCR) hash functions and the construction of a garbling scheme based on this primitive that supports free-XOR, as given in [ZRE15]. For a hash function $H_\kappa : \{0,1\}^\kappa \times \mathbb{N} \to \{0,1\}^\kappa$, consider the following two oracles:

- The oracle $O : \{0,1\}^\kappa \times \mathbb{N} \times \{0,1\} \to \{0,1\}^\kappa$ represents a random function. Note that this oracle can be simulated efficiently by lazy sampling: for each new call to O with input (x, i, b) a κ-bit string is sampled uniformly at random and assigned as the oracle output; each time O is queried with a previously queried input, the same output is delivered.
- The CCR Oracle $C_{H,\Delta} : \{0,1\}^\kappa \times \mathbb{N} \times \{0,1\} \to \{0,1\}^\kappa$ represents a function parametrized by H_κ and a κ-bit value $\Delta \leftarrow \{0,1\}^{\kappa-1}||1$ sampled uniformly at random. For each input (x, i, b), this oracle outputs the value $H_\kappa(x \oplus \Delta, i) \oplus b\Delta$.

Given these oracles, CCR secure hash functions are defined as given below.

Definition 6 (Circular Correlation Robust Hash Function). *Let $\mathcal{H} = \{H_\kappa : \{0,1\}^\kappa \times \mathbb{N} \to \{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ be a family of hash functions. Let a 'legal' sequence of oracle queries, each of the form (x, i, b), be one in which the same value*

(x, i) is never queried with different values of b. Then $\mathcal{H}$ is circular correlation robust (CCR) if for any PPT adversary A *there exists a negligible function* neg *such that for* A*'s advantage it holds*

$$\mathit{Adv}_{\mathsf{A}}^{\mathit{CCR}}(\kappa) := \frac{1}{2}\left|\Pr_{\Delta}\left[\mathsf{A}^{C_{H,\Delta}}(1^{\kappa}) = 1\right] - \Pr_{O}\left[\mathsf{A}^{O}(1^{\kappa}) = 1\right]\right| < \mathsf{neg}(\kappa).$$

Algorithms 3 and 4 detail the garbling scheme presented in [ZRE15], which can be proven selectively secure based on CCR security of the underlying hash function.

Algorithm 3 Garbling Scheme Γ^{H} for circuits with XOR and AND gates

```
1:  procedure $\mathsf{Gb}(1^\kappa, \mathbf{C})$
2:      initialize $\tilde{\mathbf{C}} = []$, $\mathbf{e} = []$ and $\mathbf{d} = []$
3:      sample $\Delta \leftarrow \{0,1\}^{\kappa-1}||1$
4:      for every $i \in [n]$ do
5:          sample $\mathsf{L}_i^0 \leftarrow \{0,1\}^\kappa$ and set $\mathsf{L}_i^1 = \mathsf{L}_i^0 \oplus \Delta$
6:          set $\mathbf{e}[i] = \mathsf{L}_i^0$
7:      end for
8:      for each $g \in [q]$ do
9:          parse $g$th gate as $(A, B, C, f_g)$
10:         if $f_g == \mathsf{XOR}$ then
11:             set $\mathsf{L}_C^0 = \mathsf{L}_A^0 \oplus \mathsf{L}_B^0$ and $\mathsf{L}_C^1 = \mathsf{L}_C^0 \oplus \Delta$
12:         else
13:             $k_g^0 = 2g - 1$, $k_g^1 = 2g$, $p_a = \mathsf{lsb}(\mathsf{L}_A^0)$, $p_b = \mathsf{lsb}(\mathsf{L}_B^0)$
14:             $G_g^0 = H(\mathsf{L}_A^0, k_g^0) \oplus H(\mathsf{L}_A^0 \oplus \Delta, k_g^0) \oplus p_b\Delta$
15:             $G_g^1 = H(\mathsf{L}_B^0, k_g^1) \oplus H(\mathsf{L}_B^0 \oplus \Delta, k_g^1) \oplus \mathsf{L}_A^0$
16:             $\mathsf{L}_C^0 = H(\mathsf{L}_A^0 \oplus p_a\Delta, k_g^0) \oplus H(\mathsf{L}_B^0 \oplus p_b\Delta, k_g^1) \oplus p_a p_b \Delta$
17:             set $\tilde{\mathbf{C}}[g] = (G_g^0, G_g^1)$
18:         end if
19:     end for
20:     for each $j \in [m]$ do
21:         set $\mathbf{d}[j] = \mathsf{lsb}(\mathsf{L}_j^0)$
22:     end for
23:     return $(\tilde{\mathbf{C}}, (\Delta, \mathbf{e}), \mathbf{d})$
24: end procedure
25:
26: procedure $\mathsf{En}((\Delta, \mathbf{e}), \mathbf{x})$
27:     initialize $\tilde{\mathbf{x}} = []$
28:     for every $i \in [n]$ do
29:         set $\tilde{\mathbf{x}}[i] = \mathbf{e}[i] \oplus \mathbf{x}[i]\Delta$
30:     end for
31:     Return $\tilde{\mathbf{x}}$
32: end procedure
33:
```

Algorithm 4 Algorithms to Evaluate the Garbling in $\mathit{\Gamma}^H$

```
procedure $\mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}})$
    initialize $\mathbf{y} = []$
    for each $g \in [q]$ do
        $\mathsf{L}_A, \mathsf{L}_B \leftarrow$ active labels associated with the input wires of $g$th gate
        if $f_g == \mathsf{XOR}$ then
            $\mathsf{L}_C = \mathsf{L}_A \oplus \mathsf{L}_B$
        else
            $k_g^0 = 2g - 1$, $k_g^1 = 2g$, $s_a = \mathsf{lsb}(\mathsf{L}_A)$, $s_b = \mathsf{lsb}(\mathsf{L}_B)$
            $\mathsf{L}_C = H(\mathsf{L}_A, k_g^0) \oplus H(\mathsf{L}_B, k_g^1) \oplus s_a G_g^0 \oplus s_b(G_g^1 \oplus \mathsf{L}_A)$
        end if
        if $C$ is a circuit output wire then
            $\mathbf{y}[C] = \mathsf{L}_C$
        end if
    end for
    return $\mathbf{y}$
end procedure

procedure $\mathsf{De}(\mathbf{y}, \mathbf{d})$
    initialize $\mathbf{y}' = []$
    for $j \in [m]$ do
        $\mathbf{y}'[j] = \mathbf{d}[j] \oplus \mathsf{lsb}(\mathbf{y}[j])$
    end for
    return $\mathbf{y}'$
end procedure
```

3 Lower Bounds on Loss in Adaptive Security of Garbling from LIN-RK-KDM-Secure Encryption

In this section, we prove that adaptive security of the garbling scheme $\mathit{\Gamma}^{(\cdot)}$ from [App16] cannot be reduced to LIN-RK-KDM security of the underlying SKE scheme. To be specific, we prove that any fully black-box security reduction R proving adaptive security of $\mathit{\Gamma}^{(\cdot)}$ based on LIN-RK-KDM security of the underlying SKE must incur an exponential loss in security. To formally state the result in Theorem 3, in Definition 7 we formally define fully black-box security reduction for the setting of Theorem 3.

Definition 7. *A **fully black-box security reduction** for Applebaum's garbling construction $\mathit{\Gamma}^{(\cdot)}$ from [App16]* based on LIN-RK-KDM security *of SKE (Definition 5) is a PPT oracle machine* R *such that for every (possibly inefficient) implementation E of SKE and every (possibly inefficient) adversary A, if A^E breaks the PRIV-IND-security of $\mathit{\Gamma}^E$ with non-negligible advantage, then $\mathsf{R}^{A,E}$ breaks the LIN-RK-KDM-security of E with non-negligible advantage. Denoting the advantage of A^E by $\epsilon_A(\kappa)$, the runtime of* R *by $t_{\mathsf{R}}(\kappa)$,*[6] *and the ad-*

[6] Assuming any fixed model of computation M, we assume each oracle call costs an oracle-aided machine in M one computational step.

vantage of $\mathsf{R}^{A,E}$ *by* $\epsilon_{\mathsf{R}}(\kappa)$*, respectively, we define the security loss for* $\mathsf{R}^{A,E}$ *as* $t_{\mathsf{R}}(\kappa)\cdot\epsilon_A(\kappa)/\epsilon_{\mathsf{R}}(\kappa)$*. The security loss of* R *is then defined as the maximal security loss of* $\mathsf{R}^{A,E}$ *over all implementations* E *of SKE and adversaries* A*.*

The following theorem now proves that any fully black-box security reduction for the garbling construction Γ^E of [App16] incurs an exponential security loss. Note that — in contrast with the lower bounds for Yao's garbling [KKPW21b] — this lower bound holds even for bounded-depth circuits.

Theorem 3. *Let* $\Gamma^{(\cdot)}$ *be the garbling scheme from [App16] (Algorithms 1 and 2). For any fully black-box security reduction* R*, there exists a family of (inefficient) deterministic adversaries* $A_{\mathcal{I}}$ *and a family of ideal SKE schemes* $\mathcal{E}$ *such that for an adversary* A_I *selected uniformly at random from* $A_{\mathcal{I}}$ *(i.e.* $I \leftarrow \mathcal{I}$*) and an SKE scheme* E *selected uniformly at random from* $\mathcal{E}$*,* A_I^E *breaks PRIV-IND security of* Γ^E *with a non-negligible probability, but for large enough* κ

$$\mathsf{Adv}^{\mathsf{LIN}}_{\mathsf{R}^{E,A_I^E}}(\kappa) \leq 2^{-\Omega(\kappa)}.$$

This holds even when $A_{\mathcal{I}}$ *is restricted to querying* NC^1 *circuits.*

To prove the theorem, we describe a family $\mathcal{E}$ of ideal encryption schemes E (Section 3.1) and an (inefficient) adversary family $A_{\mathcal{I}} = \{A_I\}_{I\in\mathcal{I}}$ that when given oracle access to E breaks PRIV-IND security of Γ^E by brute-force (Section 3.2), such that $E \leftarrow \mathcal{E}$ remains LIN-RK-KDM secure in the presence of oracle $A_I^E \leftarrow A_{\mathcal{I}}^E$. Together with the positive results of [App16], this implies that in the presence of oracle $A_I^E \leftarrow A_{\mathcal{I}}^E$ the garbling scheme Γ^E is selectively secure, i.e. sPRIV-IND secure, but *not* adaptively secure, i.e. *not* PRIV-IND secure.

3.1 Ideal Encryption E

We define the ideal encryption scheme $E = (\mathsf{Enc}, \mathsf{Dec})$ as follows:

- Key and message spaces for security parameter κ are $\mathcal{K} = \mathcal{M} = \{0,1\}^\kappa$.
- Enc is defined through an injective expanding function $f : \{0,1\}^{3\kappa} \to \{0,1\}^{4\kappa}$. On input a key k and a message m, the algorithm Enc first samples $\mathsf{r} \leftarrow \{0,1\}^\kappa$ uniformly at random and then outputs $f(\mathsf{k},\mathsf{m},\mathsf{r})$.
- Dec on input a key $\mathsf{k} \in \{0,1\}^\kappa$ and a ciphertext $\mathsf{ct} \in \{0,1\}^{4\kappa}$ brute-force searches for $\mathsf{m},\mathsf{r} \in \{0,1\}^\kappa$ such that $\mathsf{ct} = f(\mathsf{k},\mathsf{m},\mathsf{r})$. If no such m,r exist, Dec outputs $\perp$. If Dec is called on a key $\mathsf{k} = \perp$, then it outputs $\perp$.

Note, since the function f is injective, for each ciphertext ct there exists a unique key $\mathsf{k} \in \{0,1\}^\kappa$ such that $\mathsf{Dec}(\mathsf{k},\mathsf{ct}) \neq \perp$. We define the following functions $\mathsf{KEx}, \mathsf{MEx} : \{0,1\}^{4\kappa} \to \{0,1\}^\kappa$:

$$\mathsf{KEx}(\mathsf{ct}) := \begin{cases} \mathsf{k} \text{ such that } \mathsf{Dec}(\mathsf{k},\mathsf{ct}) \neq \perp & \text{if such } \mathsf{k} \text{ exists} \\ \perp & \text{else} \end{cases}$$

$$\mathsf{MEx}(\mathsf{ct}) := \mathsf{Dec}(\mathsf{KEx}(\mathsf{ct}),\mathsf{ct}).$$

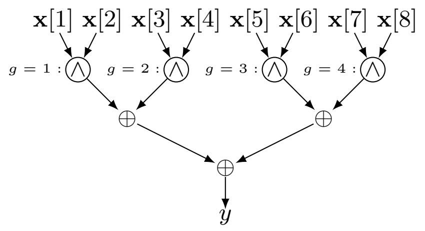

Fig. 3. The circuit $\mathbf{C}$ for $n = 2^3 = 8$. We consider the topological order on gates that numbers gates layer-by-layer.

We define the family of SKE schemes $\mathcal{E}$ as the family of SKE schemes E defined as above through functions f from the family of injective functions with domain $\{0,1\}^{3\kappa}$ and codomain $\{0,1\}^{4\kappa}$. Now, if f is sampled uniformly at random from the latter family, it is easy to see that E is exponentially LIN-RK-KDM secure (i.e. any PPT adversary in the LIN-RK-KDM security game has exponentially small advantage).

3.2 Adversarial Strategy $A_{\mathcal{I}}$

Each adversary A_I in our family of adversaries $A_{\mathcal{I}}$ has oracle access to the encryption scheme E, i.e. to both oracles Enc and Dec, which we simply denote by A_I^E. For ease of exposition, we first describe a randomised A which is sufficient for non-rewinding reductions that cannot control the adversary's randomness. The general adversary A_I, described in the end, is obtained by a simple modification of A: instead of randomly sampling, the inputs are generated deterministically using a $t_{\mathsf{R}}(\kappa)$-wise independent hash function with key I.

In the following, for security parameter κ, let $n = n(\kappa) \geq \kappa$ be a power of 2, i.e. $n = 2^d$ for some $d \in \mathbb{N}$ of order $\log(\kappa)$. When invoked on a security parameter κ, A^E first sends a circuit $\mathbf{C} : \{0,1\}^n \to \{0,1\}$ that implements the function $\{0,1\}^n \to \{0,1\}$ with $n = 2^d$ that maps $\mathbf{x} \in \{0,1\}^n$ to

$$y \;:=\; \left|\left\{g \in [n/2] \mid \mathbf{x}[2g-1] = \mathbf{x}[2g] = 1\right\}\right| \pmod 2$$

as depicted in Fig. 3.[7] A^E receives a garbled circuit $\tilde{\mathbf{C}}$ in return, and then it challenges the reduction on inputs $(\mathbf{x}_0, \mathbf{x}_1)$ sampled as

$$\mathbf{x}_1 \leftarrow \{0,1\}^n, \quad \mathbf{x}_0 \leftarrow \{\mathbf{x} \in \{0,1\}^n \mid \mathbf{C}(\mathbf{x}) = \mathbf{C}(\mathbf{x}_1)\} \setminus \{\mathbf{x}_1\}.$$

In response, it receives a garbled input $\tilde{\mathbf{x}}$ and decoding information $\mathbf{d}$. A^E outputs 1 if and only if the following checks 1–5 pass:

[7] Looking ahead, we do not really exploit any properties of the bottom layers of the circuit in our argument (and therefore the choice of XOR gates is arbitrary). But it is crucial that the first layer comprises of AND gates: the asymmetry of the garbling/gate table of AND gates will be exploited by the adversary to construct the logical map from keys to bits.

1. $\tilde{\mathbf{C}}, \tilde{\mathbf{x}}$ have the form $\tilde{\mathbf{C}} = (G_g)_{g\in[n/2]}$ with $G_g[b,c] = (G_g[b,c]^0; G_g[b,c]^1)$ and $G_g[b,c]^d$ ciphertexts (i.e. in the image of Enc) for all $g \in [n/2]$, $b,c,d \in \{0,1\}$, and $\tilde{\mathbf{x}} \in \mathcal{K}^n$.
2. $\tilde{\mathbf{C}}, \mathbf{d}$ correctly evaluates on $\tilde{\mathbf{x}}$, i.e. $\mathsf{De}(\mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}}), \mathbf{d}) = \mathbf{C}(\mathbf{x}_0)$.
3. $\tilde{\mathbf{C}}$ is properly formed w.r.t. input keys, i.e. for all $g \in [n/2]$
$$\mathsf{KEx}(G_g[0,0]^0) = \mathsf{KEx}(G_g[0,1]^0), \quad \mathsf{KEx}(G_g[1,0]^0) = \mathsf{KEx}(G_g[1,1]^0),$$
$$\mathsf{KEx}(G_g[0,0]^1) = \mathsf{KEx}(G_g[1,0]^1), \quad \mathsf{KEx}(G_g[0,1]^1) = \mathsf{KEx}(G_g[1,1]^1),$$
$$\mathsf{KEx}(G_g[0,0]^0) \oplus \mathsf{KEx}(G_g[1,0]^0) = \mathsf{KEx}(G_g[0,0]^1) \oplus \mathsf{KEx}(G_g[0,1]^1) =: \Delta$$
for some $\Delta \in \mathcal{K} = \{0,1\}^\kappa \setminus \{0^\kappa\}$
4. G_g garbles an AND gate for each $g \in [n/2]$, i.e. exists $I_g \subset \{0,1\}^2$ with $|I_g| = 3$ such that
$$\mathsf{MEx}(G_g[b,c]^0) \oplus \mathsf{MEx}(G_g[b,c]^1) =: \mathsf{L}_g^0$$
for all $(b,c) \in I_g$ and some $\mathsf{L}_g^0 \in \mathcal{M} = \{0,1\}^\kappa$, and for $(b',c') \in \{0,1\}^2 \setminus I_g$ and some $\mathsf{L}_g^1 \in \mathcal{M} = \{0,1\}^\kappa$
$$\mathsf{MEx}(G_g[b',c']^0) \oplus \mathsf{MEx}(G_g[b',c']^1) =: \mathsf{L}_g^1 \neq \mathsf{L}_g^0.$$
5. $\tilde{\mathbf{x}}$ garbles $\mathbf{x}_1$, i.e. for all $g \in [n/2]$, $b \in \{0,1\}$ and $(b',c') \in \{0,1\}^2 \setminus I_g$ as in 4
$$\mathbf{x}_1[2g-b] = [\![\tilde{\mathbf{x}}[2g-b] = \mathsf{KEx}(G_g[b',c']^{\neg b})]\!].$$

It is not difficult to see that A^E indeed breaks PRIV-IND security of Γ^E with advantage $1-2^{-\kappa}$: for an honest garbling $(\tilde{\mathbf{C}}, \tilde{\mathbf{x}}, \mathbf{d})$ of $(\mathbf{C}, \mathbf{x}_{b^*})$ all the checks 1–4 will pass whenever $\Delta \neq 0^\kappa$ (which is true with probability $1-2^{-\kappa}$).

Note that the only non-deterministic choice involved in A^E is the sampling of the inputs $(\mathbf{x}_0, \mathbf{x}_1)$. To obtain the full adversary A_I^E against general (rewinding) fully black-box security reductions, we derandomise A^E using a $t_{\mathsf{R}}(\kappa)$-wise independent hash function F (similar to [GK96]). To be specific, A_I^E evaluates $F(I,\cdot)$ on the garbled circuit $\tilde{\mathbf{C}}$ to generate the random coins r for the process used to sample $(\mathbf{x}_0, \mathbf{x}_1)$, which we denote Coll_x. See Fig. 4 for a summary of the description of A_I^E, and Lemma 1 for a formal statement that it is a valid adversary against Γ^E.

Lemma 1. *For every $I \in \mathcal{I}$ and $E \in \mathcal{E}$, A_I^E is a valid adversary against Γ^E. To be precise, for every $I \in \mathcal{I}$ and large enough $\kappa \in \mathbb{N}$, A_I^E breaks Γ^E with a probability $1-2^{-\kappa}$.*

3.3 Proof of Theorem 3

Fix a security reduction R^E for the setting of Theorem 3 indicated in Figure 2, and let $t_{\mathsf{R}}(\kappa)$ denote its running time. Note that $t_{\mathsf{R}}(\kappa)$ is an upper bound on ρ, the number of times R^E can run the adversary, including the executions by

Adversarial Strategy A_I^E

Let F be a public and efficiently computable $t_{\mathsf{R}}(\kappa)$-wise independent hash function keyed by $I \in \mathcal{I}$. Let adversary A_I^E have I hard-coded and have oracle access to $E = (\mathsf{Enc}, \mathsf{Dec}) \in \mathcal{E}$. In the PRIV-IND security game,

- For input length $n = \kappa$, A_I^E sends a circuit $\mathbf{C} : \{0,1\}^n \rightarrow \{0,1\}$ as in Figure 3 and receives a garbling $\tilde{\mathbf{C}}$.
- A_I^E computes random coins $r = F(I, \tilde{\mathbf{C}})$.
- A_I^E sends inputs $(\mathbf{x}_0, \mathbf{x}_1)$ where $\mathbf{x}_0 \neq \mathbf{x}_1$ and both are sampled using random coins r under the constraint that $\mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1)$.
- A_I^E receives $(\tilde{\mathbf{x}}, \mathbf{d})$ and outputs 1 if the following checks pass:
  1. Check if $\tilde{\mathbf{C}}$, $\mathbf{d}$ and $\tilde{\mathbf{x}}$ are of the correct size and are composed of valid elements from the ciphertext space and key space of E respectively.
  2. Check for evaluation correctness: $\mathsf{De}(\mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}}), \mathbf{d}) = \mathbf{C}(\mathbf{x}_0)$
  3. Within the garbling $\tilde{\mathbf{C}}$, extract all the encryption keys from the ciphertexts using KEx and check:
     - ciphertexts are correctly formed w.r.t. some input keys $\{(\mathsf{L}_i^0, \mathsf{L}_i^1)\}_{i \in [n]}$
     - there exists $\Delta \in \{0,1\}^\kappa$ s.t. for each input wire $i \in [n]$, $\mathsf{L}_i^0 \oplus \mathsf{L}_i^1 = \Delta$
  4. Within the garbling $\tilde{\mathbf{C}}$, extract all the messages from the ciphertexts using MEx and check:
     - ciphertexts form correct garblings of AND gates, i.e. for each gate $g \in [n/2]$, three of the associated ciphertexts encrypt the same message, the fourth a different one

     Extract the real semantic values of all the input wire labels.
  5. Check if $\tilde{\mathbf{x}}$ contains the input labels whose semantic values match $\mathbf{x}_1$

Fig. 4. Inefficient Adversarial Strategy

rewinding. To prove the theorem, we show that there exists an *efficient* "simulated" adversary $\hat{\mathsf{A}}_I^E$ such that (when $I \leftarrow \mathcal{I}$) R^E distinguishes A_I^E from $\hat{\mathsf{A}}_I^E$ with only an *exponentially-small* probability. The simulated adversary $\hat{\mathsf{A}}^E$, described below, exploits the ability to observe R's oracle queries to E and LIN-RK-KDM oracles in order to efficiently carry out the checks 1–5. It follows that for every PPT reduction R that has oracle access to the garbling adversary A_I and attempts to break LIN-RK-KDM security of E, there exists a PPT reduction R' whose advantage is only by an exponentially-small additive factor smaller than the success probability of R, and whose runtime only has a small additive overhead (incurred by simulating the efficient algorithm $\hat{\mathsf{A}}$). We note that this technique overall is reminiscent of meta-reductions [BV96], especially the proof in [GW11]. A formal description of $\hat{\mathsf{A}}_I^E$ is given below.

- $\hat{\mathsf{A}}_I^E$ initiates sets $\mathcal{Q}_E$ and $\mathcal{Q}_{\mathsf{LIN}}$, in which it stores all queries $(\mathsf{k}, \mathsf{m}, \mathsf{r}) \in \mathcal{K} \times \mathcal{M} \times \{0,1\}^\kappa$ which R makes to the $E.\mathsf{Enc}$ oracle Enc before outputting $\tilde{\mathbf{C}}$ along with the respective oracles responses ct, and all queries $(\phi, \psi) \in \Phi_{\mathsf{lin}} \times \Psi_{\mathsf{lin}}$

to the LIN-RK-KDM oracle (either $\mathsf{Real}_{\mathsf{k}^*}$ or $\mathsf{Ideal}_{\mathsf{k}^*}$ for some challenge key k^*) made before outputting $\tilde{\mathbf{C}}$ along with the responses ct, respectively. By $\mathcal{Q}_E^{\mathcal{K}}$, $\mathcal{Q}_E^{\mathcal{M}}$, $\mathcal{Q}_E^{\mathcal{C}}$ we refer to the sets of keys k, messages m and ciphertexts ct in $\mathcal{Q}_E$. Similarly, for $\mathcal{Q}_{\mathsf{LIN}}$, we refer by $\mathcal{Q}_{\mathsf{LIN}}^{\Phi}$, $\mathcal{Q}_{\mathsf{LIN}}^{\Psi}$, $\mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}$ we refer to the sets of key relations ϕ, message relations ψ and ciphertexts ct in $\mathcal{Q}_{\mathsf{LIN}}$.

- Similar to A_I^E, $\hat{\mathsf{A}}_I^E$ outputs $\mathbf{C}$ as defined in Section 3.2 as its first message.
- On receipt of a garbled circuit $\tilde{\mathbf{C}}$, $\hat{\mathsf{A}}_I^E$ samples inputs $(\mathbf{x}_0, \mathbf{x}_1)$ just like A_I^E, i.e. as $(\mathbf{x}_0, \mathbf{x}_1) := \mathbf{Coll}_x(F(I, \tilde{\mathbf{C}}))$.
- On receipt of a garbled input $\tilde{\mathbf{x}}$ and a decoding function $\mathbf{d}$, the algorithm $\hat{\mathsf{A}}_I^E$ checks whether the following are true, if any of them fails it quits and outputs 0:

1. $\tilde{\mathbf{x}} \in \mathcal{K}^n$ and all ciphertexts in $\tilde{\mathbf{C}}$ were derived as responses from either the SKE oracle or the LIN-RK-KDM oracle, i.e.
$$G_g[b,c]^d \in \mathcal{Q}_E^{\mathcal{C}} \cup \mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}} \text{ for all } g \in [n/2],\ b,c,d \in \{0,1\}.$$
2. $\tilde{\mathbf{C}}, \mathbf{d}$ correctly evaluates on $\tilde{\mathbf{x}}$, i.e. $\mathsf{De}(\mathsf{Ev}(\tilde{\mathbf{C}}, \tilde{\mathbf{x}}), \mathbf{d}) = \mathbf{C}(\mathbf{x}_0)$.
3. $\tilde{\mathbf{C}}$ is properly formed w.r.t. input keys, i.e. for all $g \in [n/2]$
$$\mathsf{KEx}(G_g[0,0]^0) = \mathsf{KEx}(G_g[0,1]^0), \quad \mathsf{KEx}(G_g[1,0]^0) = \mathsf{KEx}(G_g[1,1]^0),$$
$$\mathsf{KEx}(G_g[0,0]^1) = \mathsf{KEx}(G_g[1,0]^1), \quad \mathsf{KEx}(G_g[0,1]^1) = \mathsf{KEx}(G_g[1,1]^1),$$
$$\mathsf{KEx}(G_g[0,0]^0) \oplus \mathsf{KEx}(G_g[1,0]^0) = \mathsf{KEx}(G_g[0,0]^1) \oplus \mathsf{KEx}(G_g[0,1]^1) =: \Delta$$
for some $\Delta \in \{0,1\}^\kappa \setminus \{0^\kappa\}$.
Note that if Item 1 is true then this can be checked efficiently: Since all ciphertexts $G_g[b,c]^d$ were derived through SKE or LIN-RK-KDM oracle calls, each of them is an encryption either under a key k that was explicitly queried to the SKE oracle (hence is known), or under a key k such that its linear relation ϕ to the LIN-RK-KDM challenge key k^* was explicitly queried to the LIN-RK-KDM oracle, i.e. $\mathsf{k} = \phi(\mathsf{k}^*) = \mathsf{k}^* \oplus \Delta_\phi$ where Δ_ϕ is known. While not all keys might be known, hence also the global offset Δ might not be known, treating k^* as an unknown variable still allows to perform the check efficiently. To check $\Delta \neq 0^\kappa$, in the case that the ciphertext $G_1[b,0]^0$ for some $b \in \{0,1\}$ was derived through the SKE oracle, one can use the known key and check that decryption using that key fails for ciphertext $G_1[\neg b, 0]^0$; in the case that both ciphertexts $G_1[0,0]^0$ and $G_1[1,0]^0$ were derived through the LIN-RK-KDM oracle, one can check that the respective linear functions differ.
4. To check whether G_g garbles an AND gate for each $g \in [n/2]$, i.e. there exists $I_g \subset \{0,1\}^2$ with $|I_g| = 3$ such that
$$\mathsf{MEx}(G_g[b,c]^0) \oplus \mathsf{MEx}(G_g[b,c]^1) =: \mathsf{L}_g^0$$

for all $(b,c) \in I_g$ and some $\mathsf{L}_g^0 \in \mathcal{M}$, and for $(b',c') \in \{0,1\}^2 \setminus I_g$ and some $\mathsf{L}_g^1 \in \mathcal{M}$

$$\mathsf{MEx}(G_g[b',c']^0) \oplus \mathsf{MEx}(G_g[b',c']^1) =: \mathsf{L}_g^1 \neq \mathsf{L}_g^0,$$

check that one of the following cases is true and proceed accordingly:

a) There is a key in $\tilde{\mathbf{x}}$ that is not in $\mathcal{Q}_E^{\mathcal{K}}$.

In this case, let $i \in [n]$ be such that $\tilde{\mathbf{x}}[i] \notin \mathcal{Q}_E^{\mathcal{K}}$ and $g \in [n/2]$, $d \in \{0,1\}$ such that $i = 2g-1+d$. By check 2, it must hold $\mathsf{Dec}(\tilde{\mathbf{x}}[i], G_g[b,c]^d) \neq \perp$ for some $b,c \in \{0,1\}$. By Item 1, one of the following must hold:

- $G_g[\neg b, \neg c]^d \in \mathcal{Q}_E^{\mathcal{C}}$. In this case, there exist $\mathsf{k},\mathsf{m},\mathsf{r}$ such that $((\mathsf{k},\mathsf{m},\mathsf{r}), G_g[\neg b,\neg c]^d) \in \mathcal{Q}_E$ and one can extract Δ from Item 3 as $\Delta := \mathsf{k} \oplus \tilde{\mathbf{x}}[i]$.
- $G_g[\neg b, \neg c]^d \in \mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}$. In this case, let $\phi_i, \psi_i, \phi_i', \psi_i'$ be such that $((\phi_i,\psi_i), G_g[b,c]^d)$, $((\phi_i',\psi_i'), G_g[\neg b,\neg c]^d) \in \mathcal{Q}_{\mathsf{LIN}}$ and $\phi_i(\mathsf{k}) := \mathsf{k} \oplus \Delta_i$, $\phi_i'(\mathsf{k}) := \mathsf{k} \oplus \Delta_i'$. Then one can extract Δ from Item 3 as $\Delta := \Delta_i \oplus \Delta_i'$.

One can now check well-formedness of every gate $g \in [n/2]$ efficiently by decrypting the ciphertexts in G_g using keys $\tilde{\mathbf{x}}[2g-1]$, $\tilde{\mathbf{x}}[2g-1] \oplus \Delta$, $\tilde{\mathbf{x}}[2g]$, $\tilde{\mathbf{x}}[2g] \oplus \Delta$.

b) All keys in $\tilde{\mathbf{x}}$ are in $\mathcal{Q}_E^{\mathcal{K}}$ and there exists a key $\mathsf{k} \in \mathcal{Q}_E^{\mathcal{K}}$ and $g \in [n/2]$, $b,c,d \in \{0,1\}$ such that $\mathsf{Dec}(\mathsf{k}, G_g[b,c]^d) \neq \perp$ and $\mathsf{k} \neq \tilde{\mathbf{x}}[2g-1+d]$.

In this case, if Items 2 and 3 hold, then one can compute Δ from Item 3 as $\Delta := \mathsf{k} \oplus \tilde{\mathbf{x}}[2g-1+d]$. One can now check well-formedness of every gate $g \in [n/2]$ efficiently by decrypting the ciphertexts in G_g using keys $\tilde{\mathbf{x}}[2g-1]$, $\tilde{\mathbf{x}}[2g-1] \oplus \Delta$, $\tilde{\mathbf{x}}[2g]$, $\tilde{\mathbf{x}}[2g] \oplus \Delta$.

c) All keys in $\tilde{\mathbf{x}}$ are in $\mathcal{Q}_E^{\mathcal{K}}$ and for all keys $\mathsf{k} \in \mathcal{Q}_E^{\mathcal{K}}$ and all $g \in [n/2]$, $b,c,d \in \{0,1\}$ it holds $\mathsf{Dec}(\mathsf{k}, G_g[b,c]^d) = \perp$ or $\mathsf{k} = \tilde{\mathbf{x}}[2g-1+d]$.

In this case, $\hat{\mathsf{A}}_I^E$ simply performs the check assuming that all LIN-RK-KDM ciphertexts decrypt to $\mathbf{0} \in \{0,1\}^\kappa$. Note that if check 1 passed, then exactly half of the ciphertexts in $\tilde{\mathbf{C}}$ must be in $\mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}$, whereas the other half was encrypted under keys in $\tilde{\mathbf{x}}$. Now, since f is injective, decryption using the respective keys in $\tilde{\mathbf{x}}$ results in the same messages as MEx for all ciphertexts in $\tilde{\mathbf{C}} \cap \mathcal{Q}_E^{\mathcal{C}}$, whereas the ciphertexts in $\mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}$ are just assumed to encrypt $\mathbf{0}$. Hence, for each gate g and encryption keys $\tilde{\mathbf{x}}[2g-1], \tilde{\mathbf{x}}[2g]$, for some $b,c,d \in \{0,1\}$ one derives the correct value L_g^d whereas for the other three ciphertext pairs one arrives at $\mathsf{Dec}(\tilde{\mathbf{x}}[2g-1], G_g[b,c]^0) \oplus \mathbf{0}$, $\mathbf{0} \oplus \mathsf{Dec}(\tilde{\mathbf{x}}[2g], G_g[b,c]^1)$ and $\mathbf{0}$. The algorithm $\hat{\mathsf{A}}_I^E$ will only pass this check if three of these messages are the same and the fourth differs.

5. $\tilde{\mathbf{x}}$ garbles $\mathbf{x}_1$, i.e. for all $g \in [n/2]$, $b \in \{0,1\}$, and $(b',c') \in \{0,1\}^2 \setminus I_g$ from Item 4 it holds

$$\mathbf{x}_1[2g-b] = [\![\mathsf{Dec}(\tilde{\mathbf{x}}[2g-b], G_g[b',c']^{\neg b}) \neq \bot]\!].$$

If all the checks 1–5 pass, $\hat{\mathsf{A}}$ outputs 1, else 0.

Lemma 2. *Fix any PPT algorithm* R *with run-time* t_{R}. *There exists an exponentially-slow growing function* neg *such that*

$$\left|\Pr\left[\mathsf{R}^{E,\mathsf{Real}_{k^*},A_I^E}(1^\kappa)=1\right] - \Pr\left[\mathsf{R}^{E,\mathsf{Real}_{k^*},\hat{\mathsf{A}}_I^E}(1^\kappa)=1\right]\right| \leq t_{\mathsf{R}} \cdot 2^{-n+1} + \mathsf{neg}(\kappa),$$

where the probability is over random choice of $E \leftarrow \mathcal{E}$, $I \leftarrow \mathcal{I}$ *and random coins of reduction* R *and experiment* Real_{k^*}.

Proof: Note that $\mathbf{C}$ output as the first message by $\hat{\mathsf{A}}$ and A is fixed. Moreover, $(\mathbf{x}_0, \mathbf{x}_1)$ are generated by both $\hat{\mathsf{A}}_I$ and A_I in an identical manner as $(\mathbf{x}_0,\mathbf{x}_1) := \mathsf{Coll}_x(F(I, \tilde{\mathbf{C}}))$. We will prove for any $\tilde{\mathbf{x}}, \mathbf{d}$ and an appropriate $\delta = \delta(\kappa)$

$$\left|\Pr\left[\hat{\mathsf{A}}_I^E(\mathbf{C}, \tilde{\mathbf{C}}, (\mathbf{x}_0,\mathbf{x}_1), (\tilde{\mathbf{x}},\mathbf{d}), \mathcal{Q}_E, \mathcal{Q}_{\mathsf{LIN}}) \neq A_I^E(\mathbf{C}, \tilde{\mathbf{C}}, (\mathbf{x}_0,\mathbf{x}_1), (\tilde{\mathbf{x}},\mathbf{d}))\right]\right| \leq \delta,$$

where the probability is over random choice of $E \leftarrow \mathcal{E}$, $I \leftarrow \mathcal{I}$ and random coins of reduction R and experiment Real_{k^*}. The notation here should refer to the adversary's output bit when observing the respective transcript; recall that both A_I and $\hat{\mathsf{A}}_I$ are stateless. This then implies

$$\left|\Pr\left[\mathsf{R}^{E,\mathsf{Real}_{k^*},A_I^E}(1^\kappa)=1\right] - \Pr\left[\mathsf{R}^{E,\mathsf{Real}_{k^*},\hat{\mathsf{A}}_I^E}(1^\kappa)=1\right]\right| \leq \rho \cdot \delta.$$

It is easy to see that for $\mathbf{C}, \tilde{\mathbf{C}}, (\mathbf{x}_0,\mathbf{x}_1), (\tilde{\mathbf{x}},\mathbf{d})$ and queries $\mathcal{Q}_E, \mathcal{Q}_{\mathsf{LIN}}$ with all but exponentially small probability $\mathsf{neg}(\kappa)$ the checks 1–3 made by $\hat{\mathsf{A}}_I^E$ pass if and only if also the checks 1–3 made by A_I^E pass:

– The probability that for $\tilde{\mathbf{C}}$ output by R after making queries $\mathcal{Q}_E$, $\mathcal{Q}_{\mathsf{LIN}}$, the check 1 by A_I^E passes but the check 1 by $\hat{\mathsf{A}}_I^E$ does not pass can be bounded by an exponentially small $\mathsf{neg}(\kappa)$; this is true since $E.\mathsf{Enc}$ is defined through a random expanding function f and, hence, it is exponentially unlikely to find a ciphertext $\mathsf{ct} \in \{0,1\}^{4\kappa}$ that was not derived through an oracle query. On the other hand, if the check by $\hat{\mathsf{A}}_I^E$ passes then trivially also the check by A_I^E must pass.
– Check 2 made by $\hat{\mathsf{A}}_I^E$ is just the same as check 2 made by A_I^E, thus this check cannot trigger a different outcome at all.
– Conditioned on the previous checks passing, check 3 made by $\hat{\mathsf{A}}_I^E$ is the same as check 3 made by A_I^E; thus this check cannot trigger different output either.

Now assume all the previous checks pass and all ciphertexts in $\tilde{\mathbf{C}}$ were derived through oracle queries. If for check 4 made by $\hat{\mathsf{A}}_I^E$, cases a) or b) happen, then the check is equivalent to the check made by A_I^E in Item 4, thus these cases cannot trigger different outcome. The only interesting scenario for check 4 is if case c) happens. In this case, the outputs of $\hat{\mathsf{A}}_I^E$ and A_I^E can differ if either both checks 4 and 5 made by A_I^E pass or both checks 4 and 5 made by $\hat{\mathsf{A}}_I^E$ pass. In the following we argue that this happens with probability at most 2^{-n+1}:

If check 4 made by A_I^E passes, for each $g \in [n/2]$ there exists $I_g \subset \{0,1\}^2$ with $|I_g| = 3$ such that

$$\mathsf{MEx}(G_g[b,c]^0) \oplus \mathsf{MEx}(G_g[b,c]^1) =: \mathsf{L}_g^0$$

for all $(b,c) \in I_g$ and some $\mathsf{L}_g^0 \in \mathcal{K}$, and for $(b'_g, c'_g) \in \{0,1\}^2 \setminus I_g$

$$\mathsf{MEx}(G_g[b'_g,c'_g]^0) \oplus \mathsf{MEx}(G_g[b'_g,c'_g]^1) =: \mathsf{L}_g^1 \neq \mathsf{L}_g^0.$$

Furthermore, case c) can only happen if there exists some $\mathbf{b} \in \{0,1\}^n$ such that for all $c \in \{0,1\}$ and $g \in [n/2]$:

$$G_g[\mathbf{b}[2g-1],c]^0 \in \mathcal{Q}_E^{\mathcal{C}},\ G_g[\neg\mathbf{b}[2g-1],c]^0 \in \mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}$$
$$G_g[c,\mathbf{b}[2g]]^1 \in \mathcal{Q}_E^{\mathcal{C}},\ G_g[c,\neg\mathbf{b}[2g]]^1 \in \mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}.$$

Note that whether check 4 made by A_I^E passes and the latter happens is determined by the garbled circuit $\tilde{\mathbf{C}}$, independently of the inputs $\mathbf{x}_0, \mathbf{x}_1$ and the garbled input $\tilde{\mathbf{x}}$.
Now, for c) to happen, it must hold $\tilde{\mathbf{x}} \subset \mathcal{Q}_E^{\mathcal{K}}$, hence we must have $\tilde{\mathbf{x}}[2g-1] = \mathsf{KEx}(G_g[\mathbf{b}[2g-1],c]^0)$ and $\tilde{\mathbf{x}}[2g] = \mathsf{KEx}(G_g[c,\mathbf{b}[2g]]^1)$. But on the other hand, check 5 made by A_I^E can only pass if $\mathbf{x}_1[2g-b] = [\![\tilde{\mathbf{x}}[2g-b] = \mathsf{KEx}(G_g[b'_g,c'_g]^{\neg b})]\!]$ for all $b \in \{0,1\}$. Thus, the bad event can only happen if for all $g \in [n/2]$ it holds that

$$\mathbf{x}_1[2g-1] = [\![\mathsf{KEx}(G_g[\mathbf{b}[2g-1],c]^0) = \mathsf{KEx}(G_g[b'_g,c'_g]^0)]\!],$$

$$\mathbf{x}_1[2g] = [\![\mathsf{KEx}(G_g[c,\mathbf{b}[2g]]^1) = \mathsf{KEx}(G_g[b'_g,c'_g]^1)]\!],$$

which (by Item 3) implies

$$\mathbf{x}_1[2g-1] = [\![\mathbf{b}[2g-1] = b'_g]\!] \text{ and } \mathbf{x}_1[2g] = [\![\mathbf{b}[2g] = c'_g]\!]. \tag{3}$$

Now recall that $(\mathbf{x}_0, \mathbf{x}_1) := \mathsf{Coll}_x(F(I, \tilde{\mathbf{C}}))$, where I is the key of $t_{\mathsf{R}}(\kappa)$-wise independent hash function hardcoded into A. Since R only has oracle access to A_I^E and thus $F(I,.)$, and it can invoke A_I^E at most $t_{\mathsf{R}}(\kappa)$ times, the coins $F(I, \tilde{\mathbf{C}})$ are distributed uniformly at random to R. Furthermore, when run on uniformly random coins, Coll_x samples $(\mathbf{x}_0, \mathbf{x}_1)$ with $\mathbf{x}_0 \neq \mathbf{x}_1$ uniformly at random, hence, the probability that Eq. (3) is true is at most 2^{-n}.

Now consider the case that check 4 made by $\hat{\mathsf{A}}_I^E$ passes. As case c) happens, as above, it must hold that there exists some $\mathbf{b} \in \{0,1\}^n$ such that for all $c \in \{0,1\}$ and $g \in [n/2]$:

$$G_g[\mathbf{b}[2g-1],c]^0 \in \mathcal{Q}_E^{\mathcal{C}},\ G_g[\neg\mathbf{b}[2g-1],c]^0 \in \mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}$$

$$G_g[c, \mathbf{b}[2g]]^1 \in \mathcal{Q}_E^{\mathcal{C}},\ G_g[c, \neg\mathbf{b}[2g]]^1 \in \mathcal{Q}_{\mathsf{LIN}}^{\mathcal{C}}.$$

Now, as check 4 made by $\hat{\mathsf{A}}_I^E$ passes, we must have for each $g \in [n/2]$ and encryption keys $\tilde{\mathbf{x}}[2g-1], \tilde{\mathbf{x}}[2g] \in \mathcal{Q}_E^{\mathcal{K}}$ that three of the following four messages (note, all are $\neq \bot$) are the same and the fourth differs

$$\mathsf{Dec}(\tilde{\mathbf{x}}[2g-1], G_g[\mathbf{b}[2g-1], \mathbf{b}[2g]]^0) \oplus \mathsf{Dec}(\tilde{\mathbf{x}}[2g], G_g[\mathbf{b}[2g-1], \mathbf{b}[2g]]^1),$$

$$\mathsf{Dec}(\tilde{\mathbf{x}}[2g-1], G_g[\mathbf{b}[2g-1], \neg\mathbf{b}[2g]]^0), \quad \mathsf{Dec}(\tilde{\mathbf{x}}[2g], G_g[\neg\mathbf{b}[2g-1], \mathbf{b}[2g]]^1), \quad \mathbf{0}.$$

In particular, either $G_g[\mathbf{b}[2g-1], \mathbf{b}[2g]]^0$ and $G_g[\mathbf{b}[2g-1], \mathbf{b}[2g]]^1$ encrypt the same message, or a different one. First, consider the case that the messages are the same. Then, for check 5 made by $\hat{\mathsf{A}}_I^E$ to pass we must have $\mathbf{x}_1[2g-1] \wedge \mathbf{x}_1[2g] = 0$. Now, if $G_g[\mathbf{b}[2g-1], \neg\mathbf{b}[2g]]^0$ encrypts $\mathbf{0}$, we must have $\mathbf{x}_1[2g-1] = 0$ and $\mathbf{x}_1[2g] = 1$. On the other hand, if $G_g[\mathbf{b}[2g-1], \neg\mathbf{b}[2g]]^0$ does not encrypt $\mathbf{0}$, we get $\mathbf{x}_1[2g-1] = 1$ and $\mathbf{x}_1[2g] = 0$.
Now consider the case that $G_g[\mathbf{b}[2g-1], \mathbf{b}[2g]]^0$ and $G_g[\mathbf{b}[2g-1], \mathbf{b}[2g]]^1$ encrypt a different message. Then, if $G_g[\mathbf{b}[2g-1], \neg\mathbf{b}[2g]]^0$ encrypts $\mathbf{0}$, for check 5 made by $\hat{\mathsf{A}}_I^E$ to pass we must have $\mathbf{x}_1[2g-1] \wedge \mathbf{x}_1[2g] = 1$, hence $\mathbf{x}_1[2g-1] = 1$ and $\mathbf{x}_1[2g] = 1$. On the other hand, if $G_g[\mathbf{b}[2g-1], \neg\mathbf{b}[2g]]^0$ does not encrypt $\mathbf{0}$, we get $\mathbf{x}_1[2g-1] = 0$ and $\mathbf{x}_1[2g] = 0$.
But since $\mathbf{x}_1 \in \{0,1\}^n$ is sampled using a $t_{\mathsf{R}}(\kappa)$-wise independent hash function with key $I \leftarrow \mathcal{I}$ hard-coded in $\hat{\mathsf{A}}_I^E$ and unknown to R, this implies that the probability that check 5 passes is at most 2^{-n}.

Combining all the above, we obtain a bound $\delta(\kappa) = 2^{-n+1} + \mathsf{neg}(\kappa)$ for some exponentially small $\mathsf{neg}(\kappa)$, which proves the claim. □

For the case where R interacts with the ideal LIN-RK-KDM oracle, unless R embedded ciphertexts into $\tilde{\mathbf{C}}$ that were not derived through oracle queries, $\hat{\mathsf{A}}_I^E$ behaves just the same as A_I^E, i.e. we have the following bound.

Lemma 3. *Fix any PPT algorithm* R *with run-time* t_{R}*. There exists an exponentially-slow growing function* neg *such that*

$$\left|\Pr\left[\mathsf{R}^{E,\mathsf{Ideal}_{k^*},A_I^E}(1^\kappa) = 1\right] - \Pr\left[\mathsf{R}^{E,\mathsf{Ideal}_{k^*},\hat{\mathsf{A}}_I^E}(1^\kappa) = 1\right]\right| \leq t_{\mathsf{R}} \cdot 2^{-n+1} + \mathsf{neg}(\kappa),$$

where the probability is over random choice of $E \leftarrow \mathcal{E}$*,* $I \leftarrow \mathcal{I}$ *and random coins of reduction* R *and experiment* Ideal_{k^*}*.*

Proof: This proof works similarly to the proof of Lemma 2. As the checks 1–3 are independent of the messages encrypted in $\tilde{\mathbf{C}}$ and in particular independent of the LIN-RK-KDM challenge bit, the only difference is in bounding the probability of the bad event being triggered by the checks 4 and 5. Regarding check 4, recall from the proof of Lemma 2 that the cases a) and b) either imply that both A_I^E and $\hat{A}_I^E$ reject, or they allow to extract Δ and hence all keys that were used to derive the ciphertexts in $\tilde{\mathbf{C}}$. Thus, we only have to consider the case that $\hat{\mathsf{A}}_I^E$ rejects the check 4 and case c) happens. But also in this case the check is just the same as the check 4 made by A_I^E, since the LIN-RK-KDM oracle is implemented as $\mathsf{Ideal}_{\mathsf{k}^*}$, hence always returns encryptions of $\mathbf{0} \in \{0,1\}^\kappa$. Finally, also the check 5 made by $\hat{\mathsf{A}}_I^E$ is equivalent to the check 5 made by A_I^E, which proves the claim. □

Lemmas 2 and 3 imply the following corollary.

Corollary 1. *Fix any PPT algorithm* R *with run-time* t_R. *There exists an algorithm* R' *of about the same runtime as* R *but without oracle access to* A_I^E, *and an exponentially-slow growing function* neg *such that*

$$\mathsf{Adv}^{\mathsf{LIN}}_{\mathsf{R}^{E,A_I^E}}(\kappa) \leq \mathsf{Adv}^{\mathsf{LIN}}_{\mathsf{R}'^E}(\kappa) + t_\mathsf{R}(\kappa) \cdot 2^{-n+3} + \mathsf{neg}(\kappa).$$

where the probability is over random choice of $E \leftarrow \mathcal{E}$, $I \leftarrow \mathcal{I}$ *and random coins of reduction* R *and experiments* Real_{k^*} *and* Ideal_{k^*}.

Using the fact that our encryption scheme E is exponentially LIN-RK-KDM secure, i.e. for every PPT algorithm R' it holds $\mathsf{Adv}^{\mathsf{LIN}}_{\mathsf{R}'^E}(\kappa)$ is an exponentially-slow growing negligible function in κ, implies Theorem 3 (with $n = \Theta(\kappa)$).

4 Lower Bounds on Loss in Adaptive Security of Garbling from CCR Hashing

In this section, we prove an analogous theorem to Theorem 3 for the Half Gates scheme Γ^H from [ZRE15], described in Algorithms 3 and 4. That is, we prove that adaptive security of Γ^H cannot be black-box reduced to CCR security of the underlying hash function H. To be specific, any fully black-box security reduction R proving adaptive security of Γ^H based on CCR-security of the underlying hash function H must incur an exponential loss in security.

Definition 8. *A* ***fully black-box security reduction*** *for the garbling construction* $\Gamma^{(\cdot)}$ *of [ZRE15] based on circular correlation robustness of the hash function (Definition 6) is a PPT oracle machine* R *such that for every (possibly inefficient) implementation* H *of the hash function and every (possibly inefficient) adversary* A, *if* A^H *breaks the PRIV-IND-security of* Γ^H *with non-negligible advantage, then* $\mathsf{R}^{A,H}$ *breaks the CCR security of* H *with non-negligible advantage. Denoting the advantage of* A^H *by* $\epsilon_A(\kappa)$, *the runtime of* R *by* $t_\mathsf{R}(\kappa)$ *and the advantage of* $\mathsf{R}^{A,H}$ *by* $\epsilon_\mathsf{R}(\kappa)$, *we define the security loss of* $\mathsf{R}^{A,H}$ *as* $t_\mathsf{R}(\kappa) \cdot \epsilon_A(\kappa)/\epsilon_\mathsf{R}(\kappa)$. *The security loss of* R *is then defined as the maximal security loss of* $\mathsf{R}^{A,H}$ *over all implementations* H *of hash functions and adversaries* A.

The following theorem now proves that any fully black-box security reduction for the garbling construction Γ^E of [ZRE15] incurs an exponential security loss. Note that — similar to the lower bound for Applebaum's scheme — this lower bound holds even for bounded-depth circuits.

Theorem 4. *Let* $\Gamma^{(\cdot)}$ *be the garbling scheme from [ZRE15] (Algorithms 3 and 4). For any fully black-box security reduction* R, *there exists a family of* *(inefficient) deterministic adversaries $A_{\mathcal{I}}$ and a family of ideal hash functions $\mathcal{H}$ such that for an adversary $A_I \in A_{\mathcal{I}}$ with I selected uniformly at random from $\mathcal{I}$ and any H selected uniformly at random from $\mathcal{H}$, A_I^H breaks PRIV-IND security of Γ^H with a non-negligible probability, but for large enough κ,*

$$\mathsf{Adv}^{CCR}_{\mathsf{R}^{H,A_I^H}}(\kappa) \leq 2^{-\Omega(\kappa)}.$$

This holds even when $A_{\mathcal{I}}$ is restricted to querying NC^1 circuits.

We follow a similar approach to the one we took in the setting of garbling based on RK-KDM secure SKE: we define a family $\mathcal{H}$ of ideal hash functions H and a family of (inefficient) adversaries $A_{\mathcal{I}}$ that given oracle access to H breaks PRIV-IND security of Γ^H, while $H \leftarrow \mathcal{H}$ remains CCR secure in the presence of oracle $A_I \leftarrow A_{\mathcal{I}}$.

Proof Sketch. The proof of Theorem 4 works similarly to the proof of Theorem 3, so we will only highlight the main differences.

For the hash family $\mathcal{H}$ we choose the family of functions $H : \{0,1\}^\kappa \times \mathbb{N} \to \{0,1\}^\kappa$ such that $H(\cdot, i)$ is a permutation for every $i \in \mathbb{N}$. Note that when H is sampled uniformly at random from $\mathcal{H}$, then $H(\cdot, i)$ is a random permutation, hence information-theoretically CCR secure.

For the inefficient adversary A^H, for ease of exposition, we describe a randomised algorithm A^H, which is sufficient for non-rewinding reductions that cannot control the adversary's randomness. The general adversary A_I^H is obtained by a simple modification of A^H, exactly the same as in the proof of Theorem 3: instead of randomly sampling, the inputs are generated deterministically using a $t_{\mathsf{R}}(\kappa)$-wise independent hash function with key $I \leftarrow \mathcal{I}$.

A^H sends the same circuit $\mathbf{C}$ as in Section 3, see Fig. 3, and after receiving garbled circuit $\tilde{\mathbf{C}}$, samples $\mathbf{x}_1 \leftarrow \{0,1\}^n$ uniformly at random and $\mathbf{x}_0 \neq \mathbf{x}_1$ uniform under the constraint $\mathbf{C}(\mathbf{x}_0) = \mathbf{C}(\mathbf{x}_1)$. Breaking PRIV-IND security again works by brute-force breaking CCR security of H, however we have to be a bit more careful here:

Recall that in an honest garbling $\tilde{\mathbf{C}}$, for each $g \in [n/2]$ there are two strings associated with gate g:

$$G_g^0 = H(\mathsf{L}^0_{2g-1}, 2g-1) \oplus H(\mathsf{L}^0_{2g-1} \oplus \Delta, 2g-1) \oplus p_{2g} \cdot \Delta$$

$$G_g^1 = H(\mathsf{L}^0_{2g}, 2g) \oplus H(\mathsf{L}^0_{2g} \oplus \Delta, 2g) \oplus \mathsf{L}^0_{2g-1}$$

where $p_i = \mathsf{lsb}(\mathsf{L}_i^0)$. Furthermore, an honest input garbling $\tilde{\mathbf{x}}$ of $\mathbf{x}_1$ contains exactly one of the two labels L_i^0, $\mathsf{L}_i^0 \oplus \Delta$ for each $i \in [n]$, namely the *active* label $\mathsf{L}_i^0 \oplus (\mathbf{x}_1[i] \cdot \Delta)$ for index i. Denoting the active label by L_i we have:

$$G_g^0 = H(\mathsf{L}_{2g-1}, 2g-1) \oplus H(\mathsf{L}_{2g-1} \oplus \Delta, 2g-1) \oplus ((\mathsf{lsb}(\mathsf{L}_{2g}) \oplus \mathbf{x}_1[2g]) \cdot \Delta) \quad (4)$$

$$G_g^1 = H(\mathsf{L}_{2g}, 2g) \oplus H(\mathsf{L}_{2g} \oplus \Delta, 2g) \oplus (\mathsf{L}_{2g-1} \oplus \mathbf{x}_1[2g-1] \cdot \Delta), \quad (5)$$

where we used that $\mathsf{lsb}(\Delta) = 1$.

For any garbling $(\tilde{\mathbf{C}}, \tilde{\mathbf{x}})$ that A^H receives from the (efficient) reduction R, with probability $1 - \mathsf{neg}(\kappa)$ for some exponentially small function neg, there exists at most one $\Delta \in \{0,1\}^{\kappa-1}\|1$ solving the above equations for all $g \in [n/2]$. Hence, our inefficient adversary A^H breaks PRIV-IND security of the garbling scheme by (brute-force) searching whether a solution $\Delta \in \{0,1\}^{\kappa-1}\|1$ exists; if so, it outputs $c^* = 1$, otherwise $c^* = 0$.

It is easy to see that A^H indeed breaks PRIV-IND security of Γ^H with all but exponentially small probability: If it receives an honest garbling of $(\mathbf{C}, \mathbf{x}_1)$, then clearly there exists such Δ, namely the one drawn during the garbling procedure. On the other hand, if $(\mathbf{C}, \mathbf{x}_0)$ was garbled, then – since $\mathbf{x}_0 \neq \mathbf{x}_1$ – there must exist $i \in [n]$ such that $\mathbf{x}_0[i] \neq \mathbf{x}_1[i]$. If $i = 2g-1$ for some $g \in [n/2]$, then the Δ sampled during the garbling procedure does not satisfy Equation 5 for g; if $i = 2g$ the sampled Δ does not satisfy Equation 4 for g. On the other hand, it is exponentially unlikely that a different $\Delta \in \{0,1\}^{\kappa-1}\|1$ satisfies all equations.

To argue why the adversary A^H is not useful to break CCR security of H, we have to slightly divert from the proof of Theorem 3 since (unlike for $E.\mathsf{Enc}$) the image of the hash function H is *not* sparse in the output range and the reduction could therefore "embed" arbitrary strings into $\tilde{\mathbf{C}}$ that were not derived from any query to the hash or CCR oracle. It is relatively easy to exclude this reduction strategy and argue that the output of A^H is 0 in this case with all but negligible probability. This holds because $H(\cdot, i)$ is a random permutation for every i, and thus, it is exponentially unlikely that a solution $\Delta \in \{0,1\}^{\kappa-1}\|1$ to Equations 4 and 5 exists but no pair of labels $\mathsf{L}, \mathsf{L} \oplus \Delta$ was queried to H (either explicitly through the hash oracle or implicitly through the CCR oracle).

We thus argue uselessness of A^H for breaking CCR security of H by constructing an efficient algorithm $\hat{\mathsf{A}}^H$ that simulates A^H by observing the queries R makes to its oracles as follows:

– For each $g \in [n/2]$ and each L_{2g-1} such that a query $(\mathsf{L}_{2g-1}, 2g-1)$ was made to the hash oracle (i.e. each potential active label for index $2g-1$), $\hat{\mathsf{A}}^H$ searches for a query $(\mathsf{L}, 2g-1)$ to the hash oracle whose response was

$$G_g^0 \oplus H(\mathsf{L}_{2g-1}, 2g-1) \oplus b(\mathsf{L} \oplus \mathsf{L}_{2g-1}) \text{ for some } b \in \{0,1\},$$

or a query $(\mathsf{L}_{2g-1}, 2g-1, b)$ to the CCR oracle for some $b \in \{0,1\}$ whose response was

$$G_g^0 \oplus H(\mathsf{L}_{2g-1}, 2g-1).$$

Since $H(\cdot, 2g-1)$ is a random permutation, for each $g \in [n/2]$ this search will be successful for at most one of the L_{2g-1}, one of the oracles, and one bit b (with all but negligible probability). If for some $g \in [n/2]$ no such query exists for any L_{2g-1}, then $\hat{\mathsf{A}}^H$ outputs 0. Otherwise, $\hat{\mathsf{A}}^H$ stores the L_{2g-1} and proceeds to the next step.

– For each $g \in [n/2]$ and each L_{2g} such that a query $(\mathsf{L}_{2g}, 2g)$ was made to the hash oracle, $\hat{\mathsf{A}}^H$ searches for a query $(\mathsf{L}, 2g)$ to the hash oracle whose response was

$$G_g^1 \oplus H(\mathsf{L}_{2g}, 2g) \oplus (\mathsf{L}_{2g-1} \oplus \mathbf{x}_1[2g-1] \cdot (\mathsf{L} \oplus \mathsf{L}_{2g})),$$

or a query $(\mathsf{L}_{2g}, 2g, \mathbf{x}_1[2g-1])$ to the CCR oracle whose response was

$$G_g^1 \oplus H(\mathsf{L}_{2g}, 2g) \oplus \mathsf{L}_{2g-1}.$$

Again, since $H(\cdot, 2g)$ is a random permutation, for each $g \in [n/2]$, this search will be successful for at most one of the L_{2g}, one of the oracles, and one bit b (with all but exponentially small probability). If for some $g \in [n/2]$ no such query exists, $\hat{\mathsf{A}}^H$ outputs 0. Otherwise, $\hat{\mathsf{A}}^H$ stores the L_{2g} and proceeds to the next step.

– If $\tilde{\mathbf{x}}[i] \neq \mathsf{L}_i$ for some $i \in [n]$, then $\hat{\mathsf{A}}^H$ outputs 0. Note that in this case, with all but exponentially small probability there is no solution Δ to Equations 4 and 5 for the labels in $\tilde{\mathbf{x}}$, in which case also A^H would output 0.
– If for some $i = 2g - b \in [n]$ the former type of query exists, $\hat{\mathsf{A}}^H$ sets $\Delta := \mathsf{L} \oplus \mathsf{L}_i$ and uses this to check whether $\tilde{\mathbf{C}}$ was correctly garbled and $\tilde{\mathbf{x}}$ is consistent with $\mathbf{x}_1$(by checking whether Equations 4 and 5 hold for this value Δ for all $g \in [n/2]$). If this check passes, it outputs 1, otherwise 0.
– Finally, if for all $i \in [n]$ only the latter type of query exists, intuitively, $\hat{\mathsf{A}}^H$ must have correctly guessed $\mathbf{x}_1[i]$ for all $i \in [n]$. This is true since in the CCR security experiment, for each $(\mathsf{L}_i, i) \in \{0,1\}^\kappa \times \mathbb{N}$ there can only be a query (L_i, i, b_i) for $b_i = 0$ or $b_i = 1$, but not for both. Equations 4 and 5 however depend on $\mathbf{x}_1$, which is sampled uniformly at random *after* the garbled circuit $\tilde{\mathbf{C}}$ – and hence the values G_g^0 and G_g^1 — are chosen, and with all but exponentially small probability these equations only have a solution Δ if the CCR queries were made for one unique choice of $b_i \in \{0,1\}$ that depends on (and alternates with) $\mathbf{x}[i]$.

Hence, intuitively, the only scenario where the output of $\hat{\mathsf{A}}^H$ differs from that of A^H is if R correctly guessed $\mathbf{x}_1$, which — as $\mathbf{x}_1$ was sampled uniformly random from $\{0,1\}^n$ after $\tilde{\mathbf{C}}$ was sent — happens with probability 2^{-n} for each call the reduction makes to A^H. Setting $n = \kappa$ leads to the theorem's statement.

Acknowledgements. We thank the anonymous reviewers, whose detailed reviews and constructive suggestions helped a great deal in making the paper more formal and readable. We also thank Dennis Hofheinz for patiently answering many technical questions we had in the course of the project.

References

AAB+25. Anasuya Acharya, Karen Azari, Mirza Ahad Baig, Dennis Hofheinz, and Chethan Kamath. Securely instantiating 'half gates' garbling in the standard model. Cryptology ePrint Archive, Paper 2025/281, 2025.

AIK11. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In *FOCS 2011*, pages 120–129, 2011.

AIKW13. Benny Applebaum, Yuval Ishai, Eyal Kushilevitz, and Brent Waters. Encoding functions with constant online rate or how to compress garbled circuits keys. In *CRYPTO 2013*, pages 166–184, 2013.

App16. Benny Applebaum. Garbling XOR gates "for free" in the standard model. *J. Cryptol.*, 29(3):552–576, 2016.

BBK$^+$23. Estuardo Alpirez Bock, Chris Brzuska, Pihla Karanko, Sabine Oechsner, and Kirthivaasan Puniamurthy. Adaptive distributional security for garbling schemes with $\mathcal{O}(|x|)$ online complexity. In *ASIACRYPT (1)*, pages 139–171. Springer, 2023.

BDH14. Florian Böhl, Gareth T. Davies, and Dennis Hofheinz. Encryption schemes secure under related-key and key-dependent message attacks. In *PKC 2014*, pages 483–500, 2014.

BHKO23. Cruz Barnum, David Heath, Vladimir Kolesnikov, and Rafail Ostrovsky. Adaptive garbled circuits and garbled RAM from non-programmable random oracles. *IACR Cryptol. ePrint Arch.*, page 1527, 2023.

BHR12a. Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Adaptively secure garbling with applications to one-time programs and secure outsourcing. In *ASIACRYPT 2012*, pages 134–153, 2012.

BHR12b. Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *CCS 2012*, pages 784–796, 2012.

BMR90. Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *ACM*, pages 503–513, 1990.

BP23. Luís T. A. N. Brandão and Rene Peralta. NIST first call for multi-party threshold schemes, 2023. https://csrc.nist.gov/pubs/ir/8214/c/ipd.

BV96. Dan Boneh and Ramarathnam Venkatesan. Hardness of computing the most significant bits of secret keys in diffie-hellman and related schemes. In *CRYPTO '96*, pages 129–142, 1996.

CCPS19. Harsh Chaudhari, Ashish Choudhury, Arpita Patra, and Ajith Suresh. ASTRA: high throughput 3pc over rings with application to secure prediction. In *ACM SIGSAC 2019*, pages 81–92, 2019.

CKKZ12. Seung Geol Choi, Jonathan Katz, Ranjit Kumaresan, and Hong-Sheng Zhou. On the security of the "free-xor" technique. In *TCC 2012*, pages 39–53, 2012.

CRS20. Harsh Chaudhari, Rahul Rachuri, and Ajith Suresh. Trident: Efficient 4pc framework for privacy preserving machine learning. In *NDSS 2020*. The Internet Society, 2020.

CT21. Yu Long Chen and Stefano Tessaro. Better security-efficiency trade-offs in permutation-based two-party computation. In *ASIACRYPT 2021*, pages 275–304, 2021.

CW79. J.Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979.

CWYY23. Hongrui Cui, Xiao Wang, Kang Yang, and Yu Yu. Actively secure half-gates with minimum overhead under duplex networks. In *EUROCRYPT 2023*, pages 35–67, 2023.

GK96. Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

GKW$^+$20. Chun Guo, Jonathan Katz, Xiao Wang, Chenkai Weng, and Yu Yu. Better concrete security for half-gates garbling (in the multi-instance setting). In *CRYPTO 2020*, pages 793–822, 2020.

GKWY20. Chun Guo, Jonathan Katz, Xiao Wang, and Yu Yu. Efficient and secure multiparty computation from fixed-key block ciphers. In *IEEE SP 2020*, pages 825–841, 2020.

GLNP15. Shay Gueron, Yehuda Lindell, Ariel Nof, and Benny Pinkas. Fast garbling of circuits under standard assumptions. In *CCS 2015*, pages 567–578, 2015.

GS18. Sanjam Garg and Akshayaram Srinivasan. Adaptively secure garbling with near optimal online complexity. In *EUROCRYPT (2)*, volume 10821 of *Lecture Notes in Computer Science*, pages 535–565. Springer, 2018.

GW11. Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *STOC 2011*, pages 99–108, 2011.

GYW$^+$23. Xiaojie Guo, Kang Yang, Xiao Wang, Yu Yu, and Zheli Liu. Unmodified half-gates is adaptively secure - so is unmodified three-halves. *IACR Cryptol. ePrint Arch.*, page 1528, 2023.

HJO$^+$16. Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. In *CRYPTO 2016*, pages 149–178, 2016.

HRW16. Dennis Hofheinz, Vanishree Rao, and Daniel Wichs. Standard security does not imply indistinguishability under selective opening. In *TCC 2016-B*, pages 121–145, 2016.

JKK$^+$17. Zahra Jafargholi, Chethan Kamath, Karen Klein, Ilan Komargodski, Krzysztof Pietrzak, and Daniel Wichs. Be adaptive, avoid overcommitting. In *CRYPTO 2017*, pages 133–163, 2017.

JO20. Zahra Jafargholi and Sabine Oechsner. Adaptive security of practical garbling schemes. In *INDOCRYPT 2020*, pages 741–762, 2020.

JSW17. Zahra Jafargholi, Alessandra Scafuro, and Daniel Wichs. Adaptively indistinguishable garbled circuits. In *TCC*, pages 40–71, 2017.

JW16. Zahra Jafargholi and Daniel Wichs. Adaptive security of yao's garbled circuits. In *TCC 2016-B*, pages 433–458, 2016.

KKP21. Chethan Kamath, Karen Klein, and Krzysztof Pietrzak. On treewidth, separators and yao's garbling. In *TCC 2021*, pages 486–517, 2021.

KKPW21a. Chethan Kamath, Karen Klein, Krzysztof Pietrzak, and Michael Walter. The cost of adaptivity in security games on graphs. In *TCC 2021*, pages 550–581, 2021.

KKPW21b. Chethan Kamath, Karen Klein, Krzysztof Pietrzak, and Daniel Wichs. Limits on the adaptive security of yao's garbling. In *CRYPTO 2021*, pages 486–515, 2021.

KMR14. Vladimir Kolesnikov, Payman Mohassel, and Mike Rosulek. Flexor: Flexible garbling for XOR gates that beats free-xor. In *CRYPTO 2014*, pages 440–457, 2014.

KS08. Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications. In *ICALP 2008*, pages 486–498, 2008.

LP09. Yehuda Lindell and Benny Pinkas. A proof of security of yao's protocol for two-party computation. *J. Cryptol.*, 22(2):161–188, 2009.

LWN$^+$15. Chang Liu, Xiao Shaun Wang, Kartik Nayak, Yan Huang, and Elaine Shi. Oblivm: A programming framework for secure computation. In *IEEE Symposium on Security and Privacy, SP 2015*, pages 359–376, 2015.

NPS99. Moni Naor, Benny Pinkas, and Reuban Sumner. Privacy preserving auctions and mechanism design. In *ACM-EC*, pages 129–139, 1999.

PSSW09. Benny Pinkas, Thomas Schneider, Nigel P. Smart, and Stephen C. Williams. Secure two-party computation is practical. In *ASIACRYPT 2009*, pages 250–267, 2009.

RR21. Mike Rosulek and Lawrence Roy. Three halves make a whole? beating the half-gates lower bound for garbled circuits. In *CRYPTO 2021*, pages 94–124, 2021.

RTV04. Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In *TCC 2004*, pages 1–20, 2004.

WC81. Mark N. Wegman and J.Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3):265–279, 1981.

Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FoCS*, pages 162–167, 1986.

ZRE15. Samee Zahur, Mike Rosulek, and David Evans. Two halves make a whole - reducing data transfer in garbled circuits using half gates. In *EUROCRYPT 2015*, pages 220–250, 2015.