On the (Im)Possibility of Key Dependent Encryption

Iftach Haitner[*] Thomas Holenstein[†]

Abstract

We study the possibility of constructing encryption schemes secure under messages that are chosen depending on the key k of the encryption scheme itself. We give the following separation results:

- Let $\mathcal{H}$ be the family of poly(n)-wise independent hash-functions. There exists no fully-black-box reduction from an encryption scheme secure against key-dependent inputs to one-way permutations (and also to families of trapdoor permutations) if the adversary can obtain encryptions of $h(k)$ for $h \in \mathcal{H}$.
- Let $\mathcal{G}$ be the family of polynomial sized circuits. There exists no reduction from an encryption scheme secure against key-dependent inputs to, seemingly, *any* cryptographic assumption, if the adversary can obtain an encryption of $g(k)$ for $g \in \mathcal{G}$, as long as the reduction's proof of security treats both the adversary and the function g as black box.

Keywords: Key-dependent input security, black-box separation

1 Introduction

An cryptographic primitive is key-dependent input secure, or KDI-secure for short, if it remains secure also in case where the input depends on the secret key. In the case of encryption schemes, KDI-security means that the adversary can obtain, in addition to the usual queries, encryptions of $h(k)$, where k is the key of the scheme and h is chosen by the adversary from some (hopefully large) set $\mathcal{H}$. Ideally, $\mathcal{H}$ would be the set of all functions computable by polynomial sized circuits, but constructions of encryption schemes that are secure even against much smaller classes for $\mathcal{H}$ seem to be very difficult. For example, to the best of our knowledge there exists no construction of an encryption scheme that is secure w.r.t. the class of XOR functions $h_r(k) := k \oplus r$.

In this work we study the possibility of obtaining such an encryption scheme both from one-way functions and from other hardness assumptions. In particular, we exclude different types of reductions from KDI-secure encryption scheme to different hardness assumption. Intuitively, a *black-box reduction* of a primitive P to a primitive Q is a construction of P out of Q that ignores the internal structure of the implementation of Q and just uses it as a "subroutine" (i.e., as a black-box). The reduction, following [RTV04a], is *fully-black-box* if the proof of security (showing that an adversary that breaks the implementation of P implies an adversary that breaks the implementation of Q), is also black-box (i.e., the internal structure of the adversary that breaks the implementation of P is ignored as well). Our first result shows that there is no fully-black-box reduction from KDI-secure encryption to one-way permutations, even if the KDI-security is only against the relatively small class of poly(n)-wise independent hash-functions.

When considering reduction from a KDI-secure encryption scheme it is natural to ask whether the proof of security accesses the query function h in a black-box manner as well. Our second result, however, shows that under this restriction *no* hardness assumption implies a KDI-secure encryption scheme.

[*]Weizmann Institute of Science, Rehovot, Israel. E-mail: iftach.haitner@weizmann.ac.il. Part of this work performed while at Microsoft Research, Silicon Valley Campus.

[†]Microsoft Research, Silicon Valley Campus. E-mail: thomahol@microsoft.com

1.1 Related Work

KDI security. The development of encryption secure against key-dependent inputs started by the works of Abadi and Rogaway [AR02]. They studied formal security proofs for cryptographic protocols (as described by [DY83]), and showed that these imply security by a reduction, as long as no *key-cycles* exist in the protocol, i.e., there is a partial order $\preceq$ on the keys exists such that a message depending on k_1 would only be encrypted with k_2 if $k_1 \preceq k_2$. Since this is a restriction (even though it may be a very natural one), the community became aware that it would be desirable to create encryption schemes which provide security even in the existence of such key cycles. Consequently, Black, Rogaway, and Shrimpton [BRS02] define the (possibly stronger) notion of KDI-security for symmetric encryption schemes, and show how to obtain this notion in the random-oracle model. In such a scheme, an adversary can obtain encryptions of $h(k)$ under the key k, where h is given as a circuit to an encryption oracle. Such a scheme implies the security of the scheme under cycles as well. Independently of [BRS02], a notion of circular security has been defined by Camenisch and Lysyanskaya [CL01], considering asymmetric encryption schemes as well.

Recently, Halevi and Krawczyk [HK07] generalized the notion of KDI-security to other cryptographic primitives, such as pseudorandom functions. They also coined the name KDI for this sort of security (previously, it was named *key-dependent message security*). Their results in this setting is mainly for the construction of pseudorandom functions. In addition they showed that a *deterministic* encryption scheme cannot be KDI-secure. Independently and concurrently to their work, Hofheinz and Unruh [HU08] provide private key encryption schemes secure under a limited class of KDI-attacks. The main limitation of their work is that the scheme only remains secure as long as $h(k)$ is significantly shorter than the key; also, after every application of the encryption scheme the key is updated. This makes the construction insufficient for the initial motivation of allowing key cycles in cryptographic protocols.

Black-box impossibility results. Impagliazzo and Rudich [IR89] showed that there are no black-box reductions of key-agrement protocols to one-way permutations and substantial additional work in this line followed (c.f. [GKM$^+$00, Rud88, Sim98]). Kim, Simon and Tetali [KST99] initiated a new line of impossibility results, by providing a lower bound on the *efficiency* any black-box reduction of universal one-way hash functions to one-way permutations, substantial additional work in this line followed (c.f. [GGKT05, HHRS07, HK05, Wee07]).

1.2 Contributions of this Paper

In this paper we give two impossibility results for constructions of KDI-secure private-key encryption schemes.[1] The first one is a black-box separation to one-way permutations (our result generalizes to trapdoor permutations).

THEOREM 1.1
There exists no fully-black-box reduction from an encryption scheme that is KDI-secure against every $\mathrm{poly}(n)$-wise independent family of hash functions to one-way permutations.

More exactly, for any candidate black-box construction of an encryption scheme, there exist a polynomial $p(n)$ such that the KDI-security against the family of $p(n)$-wise independent hash-functions has no black-box proof of security. As in [GT00, HHRS07], this result can be extended to (enhanced) family of trapdoor permutations by an easy modification of the proof.

Assume that the query function *itself* under which the scheme should be KDI-secure is treated by the proof of security as black-box. Moreover, the proof of security does not forward an access to the query function to a third party (the reader might refer Remark 1.3, for a proof of security without the latter property). We call such a reduction ***strongly-black-box*** and give the following result.

[1]Clearly, our results extend to the public-key case.

THEOREM 1.2 **(informal)**
There exists no reduction with strongly-black-box proof of security from a KDI encryption scheme to any secure cryptographic hardness hardness assumption.

More exactly, Theorem 1.2 excludes reductions whose proof of security is black-box and treats the query function as black-box. The construction of the encryption scheme itself, however, can be arbitrary.

REMARK 1.3
We stress that the above theorem has no conflict with the tautology that KDI-security implies KDI security. Given a KDI-secure encryption scheme $(\mathrm{Enc}, \mathrm{Dec})$, we use the following reduction to construct a KDI-secure encryption scheme $(\mathrm{Enc}', \mathrm{Dec}')$. We let Enc' and Dec' to be equal to Enc and Dec, where in the proof of security we use an adversary A that breaks the KDI-security of $(\mathrm{Enc}', \mathrm{Dec}')$ for violating the KDI-security of $(\mathrm{Enc}, \mathrm{Dec})$ as follows. On security parameter n, we ask A for its query function h, then forward h to the KDI oracle to get $c = \mathrm{Enc}(k, h(k))$ and return c back to A. Finally, we return the same answer as A does. Clearly the above reduction is fully-black-box. The proof of security, however, is *not* strongly-black-box, since the handle to h is forwarded to a third party (i.e., the KDI oracle).

Nevertheless, we think that our result shows that it should be very difficult to construct a KDI-secure encryption scheme, since most natural hardness assumptions do not have a clear place where the strong black-box property fails.

1.3 Our Technique

In the proof of both our results, we are using the same oracle, Breaker, that helps us to break the KDI-security of every encryption scheme. Let $(\mathrm{Enc}, \mathrm{Dec})$ be some fixed encryption scheme. On input (h, c), where h is some length doubling function and c is a ciphertext, Breaker considers all possible keys, and returns the first key k for which $\mathrm{Dec}(k, c) = h(k)$, or $\perp$ if no such key exists. It is not hard to show that $(\mathrm{Enc}, \mathrm{Dec})$ is not KDI-secure in the presence of Breaker. Therefore, our impossibly results follow if Breaker does not help to violate the underlying hardness assumption. For this, we need to assume that Breaker is only called with functions h which are chosen uniformly from the respective set of query functions. We ensure this by restricting the functions h for which Breaker performs the above computation to one which is randomly chosen (and then give the adversary access to it). Under this restriction, Breaker does not help breaking the assumption. Proving this is our main technical contribution (note that Breaker cannot be implemented efficiently) and we prove it differently in each of our separation results.

One-way permutations. Let π be a random permutation and let $(\mathrm{Dec}^{\pi}, \mathrm{Enc}^{\pi})$ be a candidate of a KDI-secure encryption scheme with black-box construction, given π as the one-way permutation. We find a polynomial $p(n)$ (which depends on $(\mathrm{Dec}^{\pi}, \mathrm{Enc}^{\pi})$) and use a family of $p(n)$-wise independent hash-functions as query functions. Intuitively, if Breaker helps an algorithm A inverting π, then in such a call the behavior of Breaker must be very different for a large number of potential preimages of y. We show, however, that for all but a negligible fraction of the functions h, the behavior of Breaker will be the same for most possible preimages of y, no matter how the ciphertext is chosen.

Arbitrary assumptions. In this case h is a completely uniform random function chosen by the system. The idea is that for such h, all calls to Breaker are very likely to be answered with $\perp$, as for any fixed $k \in \{0,1\}^t$, the probability that $\mathrm{Dec}(k, c) = h(k)$ is roughly 2^{-2t}. Now, this somewhat naive intuition is actually false, as it can fail in the following way: A picks a key k' itself, queries $h(k')$, encrypt this with k' itself, and gives the resulting ciphertext to Breaker. We prove, however, that this is essentially the only way in which the above intuition fails. Thus, instead of calling Breaker, A can as well check the keys on which h was previously queried, which can be done efficiently.

Finally, we show that h itself cannot help in breaking the initial hardness assumption. Hence, if there is a reduction with strongly-black-box proof of security from a KDI encryption scheme to a given hardness assumption, the hardness assumption is not secure.

1.4 Paper Organization

We present the notations and formal definitions used in this paper in Section 2. In Section 3 we give the separation from one-way permutations, and in Section 4 we give the separation from any hardness assumption. Finally, in Section 5 we consider the question whether the techniques used in this paper are also useful for proving impossibility result w.r.t. other KDI-secure cryptographic primitives.

2 Preliminaries

2.1 Notation

We denote the concatenation of two strings x and y by $x \circ y$. If X is a random variable taking values in a finite set $\mathcal{U}$, then we write $x \leftarrow \mathcal{U}$ to indicate that x is selected according to the uniform distribution over X. We often use probabilities where we choose an oracle π from some uncountable set of oracles at random. To make these well defined, one has to define the uniform measure on the oracles, which can be done by mapping an oracle to a real $r \in [0, 1)$ (for example by interpreting the truth table of the oracle as binary representation of r), and then using Lebesgue-measure. One can verify that with this definition indeed all random variables which occur in this paper are measurable.

2.2 Many-wise Independence

DEFINITION 2.1 **(s-wise independent random variables)**
Let $X_1, \ldots, X_n$ be random variables defined over $\mathcal{U}$. $X_1, \ldots, X_n$ are s-wise independent for some $s \in N$ if every sequence $(X_{i_1}, \ldots, X_{i_s})$ of random variables is uniformly distributed over $\mathcal{U}^s$.

DEFINITION 2.2 **(s-wise independent hash functions)**
Let $\mathcal{H}$ be a family of functions mapping $\{0,1\}^\ell$ to $\{0,1\}^m$, $s \in \mathbb{N}$. We say that $\mathcal{H}$ is an efficient family of s-wise independent hash functions (following [CW79]) if the following hold:[2]

Samplable. $\mathcal{H}$ is samplable in time $\text{poly}(s, \ell, m)$.

Efficient. There exists an algorithm that given $x \in \{0,1\}^\ell$ and a description of $h \in \mathcal{H}$ outputs $h(x)$ in time $\text{poly}(s, \ell, m)$.

s-wise independence. If h is chosen at random from $\mathcal{H}$, the 2^ℓ random variables $h(x)$, $x \in \{0,1\}^\ell$ are s-wise independent.

It is well known ([CW79]) that there exists an efficient family of s-wise independent hash functions from ℓ-bit strings to m-bit strings for every choice of ℓ and m whose elements description size is $\mathcal{O}(s \cdot \max\{\ell, m\})$.

We use the following upper bound on the probability that many s-wise independent events occur, where each event has low probability. The usual bounds seem not strong enough in our setting, as they focus on the range where the probability of a single event is constant (for example, this seems to be the case for the bound given in [Gol01]). In our case, the probability of a single event decreases as the number of events increases, and we therefore use a different bound.

[2]The first two properties, regarding the efficiency of the family, implicitly assume an ensemble of families (one family for every triple (s, ℓ, m)). For simplify of presentation, we only refer to a single family.

LEMMA 2.3
For $s, V \in \mathbb{N}$ let $B_1, \ldots, B_V$ be s-wise independent Bernoulli random variables with $\Pr[B_i = 1] \le \frac{1}{V}$. Assuming that $\alpha > s$, then $\Pr\left[\sum_{i=1}^{V} B_j \ge \alpha\right] < \frac{\log(V)}{\alpha^{s-1}}$.

Proof. The idea of the proof is to find a small family, $\mathcal{T}$, of sets of size s in $\mathcal{U} = \{1, \ldots, V\}$, such that every set in $\mathcal{U}$ of size α has a subset in $\mathcal{T}$. In this case, one can use the union bound on the sets in $\mathcal{T}$ (as explained in the proof). We first find $\mathcal{T}$.

CLAIM 2.4
There exists a family of $\alpha \log(V)(\frac{V}{\alpha})^s$ sets of size s such that the following holds. Every set $Q \subseteq \mathcal{U}$ of size $|Q| = \alpha$ has a subset $\tau \subseteq Q$ in $\mathcal{T}$.

Proof. Directly with the probabilistic method. Pick $\alpha \log(V)(\frac{V}{\alpha})^s$ sets of size s at random. The probability that a fixed set of size α has none of these sets as a subset is $(1 - (\frac{\alpha}{V})^s)^{\alpha \log(V)(\frac{V}{\alpha})^s} < e^{-\alpha \log(V)}$. Since there are $\binom{V}{\alpha} < V^\alpha = e^{\alpha \log(V)}$ subsets of size α in $\mathcal{U}$, the expected number of uncovered sets is smaller than 1, which implies that with positive probability we obtain a family as required. $\square$

Consider now a family $\mathcal{T}$ with $|\mathcal{T}| = \alpha \log(V)(\frac{V}{\alpha})^s$ of sets of size s, such that every set $Q \subseteq \mathcal{U}$ of size α has a subset in $\mathcal{T}$ (as guaranteed by Claim 2.4). If $\sum_{i=1}^{V} B_j \ge \alpha$, then there exists a set L of size α such that $B_j = 1$ for all $j \in L$. This implies that there exists a set $\tau \in \mathcal{T}$ such that $B_i = 1$ for all $i \in \tau$. By the union bound, the probability that the latter happens for some subset in $\mathcal{T}$ is at most

$$|\mathcal{T}|\left(\frac{1}{V}\right)^s \le \alpha \log(V)\left(\frac{V}{\alpha}\right)^s\left(\frac{1}{V}\right)^s = \frac{\log(V)}{\alpha^{s-1}} \ .$$

$\square$

2.3 Encryption Schemes and KDI-security

We define a (private-key) encryption scheme as a pair of an encryption and a decryption algorithm (Enc, Dec). On security parameter n, the encryption algorithm Enc gets as input a key of length $t(n)$ and a message of length m, and outputs a ciphertext of length $\ell(n, m)$. The decryption algorithm Dec, gets a key and a ciphertext and outputs the message.[3] Informally, an encryption scheme (Enc, Dec) is KDI-secure against a family of functions $\mathcal{H} \subseteq \{h : \{0,1\}^t \to \{0,1\}^*\}$, if no efficient adversary can distinguish between an oracle which correctly returns an encryption of $h(k)$, given input h, and one which returns an encryption of the all zero string of the same length. Note that if $\mathcal{H}$ contains functions which map to constants, normal encryption queries can be obtained as well.

DEFINITION 2.5 (KDI-security)
Given an encryption scheme (Enc, Dec) and a key of length t, let $Q^{\mathrm{Enc},k}$ [resp. $\widetilde{Q}^{\mathrm{Enc},k}$] be an algorithm that gets as input a function $h : \{0,1\}^t \mapsto \{0,1\}^m$, and returns $\mathrm{Enc}(k, h(k))$ [resp. $\mathrm{Enc}(k, 0^m)$] (if the schemes is randomized, it returns a random encryption). We say that (Enc, Dec) is KDI-secure for a class of functions $\mathcal{H}$, if

$$\left|\Pr_{k \leftarrow \{0,1\}^{t(n)}}[A^{Q^{\mathrm{Enc},k}}(1^n) = 1] - \Pr_{k \leftarrow \{0,1\}^{t(n)}}[A^{\widetilde{Q}^{\mathrm{Enc},k}}(1^n) = 1]\right| \ ,$$

is negligible for every efficient algorithm A that only queries functions in $\mathcal{H}$.

[3]In some definitions, a private key encryption scheme also includes a key-generation algorithm "Gen". We omit this since we are not concerned by polynomial factors, and in this case one can simply take the random coins used by Gen as private key.

2.4 Hardness Assumptions

For reductions which treat the family $\mathcal{H}$ of query-functions as black-box, we are able to prove a very strong impossibility result. In this case, we show that *no* cryptographic assumption is sufficient to guarantee the KDI-security of the scheme. In order to do this, we first define the set of cryptographic assumptions we consider. We try to be as general as possible here, while keeping the complexity of the definition on a reasonable level.

DEFINITION 2.6 (cryptographic games)
A cryptographic game is defined random system Γ that, on security parameter n interacts with an attacker A and may output a special symbol win. In case $\Gamma(1^n)$ outputs this symbol in an interaction with $A(1^n)$, we say that $\mathbf{A(1^n)} \leftrightarrow \mathbf{\Gamma(1^n)}$ ***wins***. The game is ***secure*** if $\Pr[A(1^n) \leftrightarrow \Gamma(1^n) \text{ wins}]$ is negligible for all PPT A, where the probability is over the randomness of A and Γ.

Examples: One might define the security of a one-way function f by the following game. On security parameter n, the system Γ selects a random $x \in \{0,1\}^n$ and sends $y = f(x)$ to the adversary. Γ outputs win if A outputs $x' \in f^{-1}(y)$.

To define the DDH hardness assumption one needs a bit more work. On security parameter n, the system Γ expects first a sequence of at least n ones, we denote the actual number received by α. The system Γ then sends A a description of an appropriately chosen group $\langle g \rangle$ of order $\Omega(2^n)$ and the generator g, as well as α randomly chosen triples $(g^{x_i}, g^{y_i}, g^{z_i})$, where $z_i = x_i y_i$ or a uniform random element, each with probability $\frac{1}{2}$. The attacker A wins, if the number of instances where he incorrectly predicts whether z_i was chosen independently of x_i and y_i, is at most $\frac{\alpha}{2} - \alpha^{2/3}$.

Clearly, if Γ is secure, the usual DDH hardness assumption holds, since any algorithm that breaks the DDH assumption can be used for violating the security of Γ.[4] The reverse direction is implied by the main result (i.e., Theorem 1) of [IJK07] as follows. Assume the existence of a polynomial time adversary A violating the security of Γ. Let $\varepsilon(n) \in 1/\operatorname{poly}(n)$ be such that A's winning probability is more than $\varepsilon(n)$ infinitely often and $\alpha < \frac{1}{\varepsilon(n)}$. Let $\delta = \frac{1}{2} - \varepsilon(n)$, $\gamma = \frac{1}{\alpha^{1/3}}$ and $k = \alpha$. Under the DDH assumption, [IJK07, Theorem 1] yields that the probability of A to be wrong on fewer than $T = (1-\gamma)\delta k = (1-\frac{1}{\alpha^{1/3}})(\frac{1}{2}-\varepsilon(n))\alpha > \frac{\alpha}{2} - \frac{\alpha}{2\alpha^{1/3}} - \alpha\varepsilon(n) > \frac{\alpha}{2} - \alpha^{2/3}$ of the games is bounded by $\varepsilon(n) > e^{-\Omega(\gamma^2\delta^2 k)}$, contradicting the existence of A.

2.5 Black-Box Reductions

A reduction from a primitive P to a primitive Q consists of showing that if there exists an implementation C of Q, then there exists an implementation M_C of P. This is equivalent to showing that for every adversary that breaks M_C, there exists an adversary that breaks C. Such a reduction is semi-black-box if it ignores the internal structure of Q's implementation, and it is fully-black-box (using the terminology of [RTV04b]) if it also has black-box proof of security. That is the adversary for breaking Q ignores the internal structure of both Q's implementation and of the (alleged) adversary breaking P. The following definition expands the above general discussion for the case of a fully-black-box reduction of a KDI-secure encryption scheme from a one-way permutation.

DEFINITION 2.7 (fully-black-box reduction)
A fully-black-box reduction of a KDI-secure encryption scheme from a one-way permutation consists of polynomial-time oracle-aided algorithms $(\mathrm{Enc}^{(\cdot)}, \mathrm{Dec}^{(\cdot)})$ and a polynomial-time adversary $A_{\mathrm{OWP}}^{(\cdot)}$, such that the following hold:

[4]Given an algorithm A that breaks that DDH assumption with advantage $1/p(n)$, we construct the following attacker A' for Γ. As first message A' sends $1^{p(n)^3}$ to Γ, and then applies A for each of the triples it gets back from Γ.

- If f is a permutation, then $(\mathrm{Enc}^f, \mathrm{Dec}^f)$ is an encryption scheme.
- If A_{KDI} breaks the KDI-security of the encryption scheme, then $A_{\mathrm{OWP}}^{f,A_{\mathrm{KDI}}}$ inverts the permutation with non-negligible probability.

When considering reductions from a KDI secure cryptosystem, it is natural to consider whether the proof of security accesses the challenging functions also as a black-box. We say that a proof of security from KDI cryptosystem is ***strongly-black-box*** if it treats the challenging functions also as a black-box.

DEFINITION 2.8 (strongly-black-box reduction)
A reduction from a KDI-secure encryption scheme to a cryptographic assumption Γ with strongly-black-box proof of security, consists of polynomial time oracle algorithms $(\mathrm{Enc}, \mathrm{Dec})$ and a polynomial time oracle-aided adversary $A_\Gamma^{(\cdot)}$ such that the following holds:

- $(\mathrm{Enc}, \mathrm{Dec})$ is an encryption scheme.
- For any adversary A_{KDI}^{Q} which breaks the KDI-security of $(\mathrm{Enc}, \mathrm{Dec})$, the oracle-aided adversary $A_\Gamma^{(A_{\mathrm{KDI}})}$ violates the security of Γ. Additionally, A_Γ treats the challenging functions provided by A_{KDI} as a black box.

The requirement that A_Γ treats the challenging function as black box means that it can only obtain evaluations of it at arbitrary chosen points. Furthermore, the reduction must work for *any* challenging function (not just efficiently computable ones). In particular, the reduction does not provide Γ with a description of the function.

2.6 Extending KDI-secure Encryption Schemes

The following (straightforward, but technically slightly tedious) proposition allows us to prove our impossibility results for encryption schemes which have the following additional properties:

- Perfect correctnes, i.e., it is always the case that $\mathrm{Dec}(k, \mathrm{Enc}(k, m)) = m$
- Deterministic encryption, i.e., the decryption procedure Dec does not use any randomness
- Defined for messages of arbitrary length
- Key length equals the security parameter

The last point poses no problem if the key length is larger than the security parameter by a simple redefinition. If a scheme uses a very short key (e.g. of length $t = O(\log^2(n))$), however, it seems that padding is required, and in fact we show in the proposition that such padding poses no problem either.

In the following, for a set of boolean functions $\mathcal{H} \subseteq \{h : \{0,1\}^t \to \{0,1\}\}$, we let $\mathcal{H}^* \subseteq \{h : \{0,1\}^t \to \{0,1\}^*\}$ be the set of concatenations of functions from $\mathcal{H}$. [5] Additionally, for $t' \leq t$ we let $\mathcal{H}_{|t'}$ be the set of functions which are obtained by fixing the last $t - t'$ bits to arbitrary values.

PROPOSITION 2.9
Fix a set of functions $\mathcal{H}$ and let $(\mathrm{Enc}, \mathrm{Dec})$ be an encryption scheme with key length t that is KDI-secure for $\mathcal{H}_{|t}$. Then there exists an encryption scheme $(\mathrm{Enc}_1, \mathrm{Dec}_1)$ with key length $t_1 \geq t$ that is KDI-secure for $\mathcal{H}^*$, uses $(\mathrm{Enc}, \mathrm{Dec})$ as a black-box, is always correct, has deterministic decryption algorithm and is defined for every message length. Moreover, if $(\mathrm{Enc}, \mathrm{Dec})$ has a strongly-black-box [resp. black-box] proof of security to a cryptographic game Γ, then $(\mathrm{Enc}_1, \mathrm{Dec}_1)$ has a strongly-black-box [resp. black-box] proof of security to Γ.

[5] That is, $h : \{0,1\}^t \to \{0,1\}^\ell \in \mathcal{H}^*$ iff there exists $h_1, \ldots, h_\ell \in \mathcal{H}$ such that for every input x it holds that $h(x) = (h_1(x) \circ \cdots \circ h_\ell(x))$, where $\circ$ denotes the concatenation of strings.

Proof. We start by handling the correctness and deterministic decryption parts. Let $(\mathrm{Enc}, \mathrm{Dec})$ be an encryption scheme. The scheme $(\mathrm{Enc}_0, \mathrm{Dec}_0)$ uses $(\mathrm{Enc}, \mathrm{Dec})$ as a subroutine and works as follows: on inputs (k, m), Enc_0 chooses, in addition to r_{Enc}, the random coins needed by Enc, also the random coins to be used by Dec, r_{Dec}. It then checks whether $\mathrm{Dec}(r_{\mathrm{Dec}}, k, \mathrm{Enc}(r_{\mathrm{Enc}}, k, m)) = m$. If this holds, then $\mathrm{Enc}_0(k, m)$ outputs $0 \circ \mathrm{Enc}(k, m) \circ r_{\mathrm{Dec}}$, otherwise it outputs $1 \circ m$ (where $\circ$ denotes the concatenation of strings). On ciphertext $c = (0 \circ c \circ r_{\mathrm{Dec}})$, algorithm $\mathrm{Dec}_0(k, c)$ outputs $\mathrm{Dec}(r_{\mathrm{Dec}}, k, c)$, where on ciphertext $c = (1 \circ m)$ it outputs m.

Having the above, we define $(\mathrm{Enc}_1, \mathrm{Dec}_1)$ as follows. Given a message to encrypt of arbitrary length, we partition the message into blocks of the appropriate size (while padding the last block) and encrypt each of the blocks using Enc_0. In the decryption we do the same using Dec_0. Additionally, if $t_1 > t$, Enc_1 chooses $t_1 - t$ additional key-bits at random. All in all, $(\mathrm{Enc}_1, \mathrm{Dec}_1)$ has a deterministic encryption algorithm and is always correct.

We construct an adversary $A^{(\cdot)}$ for $(\mathrm{Enc}, \mathrm{Dec})$ given an adversary for $(\mathrm{Enc}_1, \mathrm{Dec}_1)$ as follows. Let A_1 be an adversary for $(\mathrm{Enc}_1, \mathrm{Dec}_1)$. The adversary A^{A_1} first picks $t_1 - t$ additional key bits, and then emulates A_1. Every time A_1 asks for an encryption of a function $h \in \mathcal{H}^*$, A^{A_1} decomposes h into its components $h_i \in \mathcal{H}$. If $t_1 > t$ it computes the respective functions in $\mathcal{H}_{|t}$, and then obtains each of the encryptions from the given KDI-encryption oracle of $(\mathrm{Enc}, \mathrm{Dec})$. Note that this decomposition can be done in a black-box manner, by considering for each h_i only the relevant part in the output of h. Then, A^{A_1} prefixes each answer with a 0 (to denote that the decryption is correct), adds uniformly chosen randomness for the decryption, concatenates the answers and returns the resulting ciphertext to A_1. One checks that with high probability the distribution of the resulting ciphertext is as required. Once A_1 gives a prediction, A^{A_1} outputs this prediction.

Let $A_\Gamma^{(\cdot)}$ be the adversary, guaranteed by the black-box proof of security of $(\mathrm{Enc}, \mathrm{Dec})$, that violates the security of Γ given an adversary for $(\mathrm{Enc}, \mathrm{Dec})$. It is clear that $A_\Gamma^{A^{(\cdot)}}$ is a black-box proof of security for $(\mathrm{Enc}_1, \mathrm{Dec}_1)$. Moreover, assuming that $(\mathrm{Enc}, \mathrm{Dec})$ has a strongly-black-box proof of security (and that $A_\Gamma^{(\cdot)}$ is the adversary guaranteed by this proof), it is easy to verify that $A_\Gamma^{A^{(\cdot)}}$ is a strongly-black-box proof of security for $(\mathrm{Enc}_1, \mathrm{Dec}_1)$. □

3 One-Way Permutations

In this section we prove Theorem 1.1. Let $(\mathrm{Enc}^{(\cdot)}, \mathrm{Dec}^{(\cdot)})$ be an encryption scheme with oracle access to a one-way permutation. By Proposition 2.9, we can assume that the encryption scheme is always correct, has deterministic decryption algorithm, and is defined on messages of any polynomial length. We use the following notations. For a security parameter t (which for simplicity and using Proposition 2.9, we assume to be equal to the key length) let $\ell(t)$ be the length of an encryption of a message of length $2t$. Furthermore, let $\mathcal{H}_t$ be a family of $(\ell(t) + t)$-wise independent hash functions from $\{0,1\}^t$ to $\{0,1\}^{2t}$, with polynomial description size.[6] Let $\pi = \{\pi_n : \{0,1\}^n \to \{0,1\}^n\}_{n \in \mathbb{N}}$ be a family of functions. In order to prove Theorem 1.1, we use oracle access to the following inefficient algorithm $\mathrm{Breaker}^\pi$.

ALGORITHM 3.1

The algorithm $\mathrm{Breaker}^\pi$.

Randomness: A uniformly chosen family $\{h_t \in \mathcal{H}_t\}_{t \in \mathbb{N}}$.

Input: A pair (t, c), where $c \in \{0,1\}^{\ell(t)} \cup \{\star\}$, $\star$ being a special symbol.

Output: If $c = \star$ return the description of h_t. Otherwise, return the smallest $k \in \{0,1\}^t$ such that $\mathrm{Dec}^\pi(k, c) = h_t(k)$, or $\perp$ if no such k exists.

[6]Note that a s-wise independent hash-function remains s-wise independent after padding.

In Section 3.1 we show that $\mathrm{Breaker}^{\pi}$ does not help inverting a random π. In Section 3.2, we show that $\mathrm{Breaker}^{\pi}$ breaks the KDI security of $(\mathrm{Enc}, \mathrm{Dec})$ and prove Theorem 1.1.

3.1 Breaker Does not Invert Random Permutations

We prove the following upper bound on the probability that an algorithm with access to Breaker inverts a random permutation. In the following let $\mu_A(n)$ [resp. $\mu_{\mathrm{Dec}}(t)$] be an upper bound on the total length of all oracles queries $A(y)$ [resp. Dec] does for $y \in \{0,1\}^n$ [resp. $(k, c) \in \{0,1\}^t \times \{0,1\}^{\ell(t)}$]. We assume without loss of generality that both upper bounds are monotonically increasing, that $\mu_{\mathrm{Dec}}(t) \geq t + \ell(t)$, and $\mu_A(n) \geq n$.

LEMMA 3.2
Let A be an adversary with oracle access to π and Breaker. For any $y \in \{0,1\}^n$:

$$\mathsf{Pr}_{\pi}[A^{(\pi,\mathrm{Breaker})}(y) = \pi^{-1}(y)] < 3\mu_A(n)\left(2^{-\mu_{\mathrm{Dec}}^{-1}(n)} + \mu_{\mathrm{Dec}}(\mu_A(n))^2 2^{-n}\right) \quad ,$$

where $\mu_{\mathrm{Dec}}^{-1}(n) := \min\{t \in \mathbb{N} \ : \ \mu_{\mathrm{Dec}}(t) \geq n\}$.

In particular, if μ_A and μ_{Dec} are both polynomials, Lemma 3.2 yields that $\mathsf{Pr}_{\pi}[A^{(\pi,\mathrm{Breaker})}(y) = \pi^{-1}(y)]$ is negligible. A lemma which is roughly equivalent to Lemma 3.2 can be proven with the technique introduced by Gennaro and Trevisan [GT00], we do this in Appendix A. Here, we will use a different technique that is more similar to the one used by Simon [Sim98].

The main idea is to study what happens if π is modified slightly by mapping a second, randomly chosen element to y. We show that such a change will likely go unnoticed by the adversary, and he will not find the new preimage. After the change, however, both preimages of y are equally likely to be the original one, so the adversary cannot have found the original one either.[7] For a given function π and two strings $x^*, y \in \{0,1\}^n$, we define the function $\pi_{|x^* \mapsto y}$ as

$$\pi_{|x^* \mapsto y}(x) := \begin{cases} y & \text{if } x = x^*, \\ \pi(x) & \text{otherwise.} \end{cases}$$

We assume that all calls to $\mathrm{Dec}^{\pi_{|x^* \mapsto y}}$ and $\mathrm{Breaker}^{\pi_{|x^* \mapsto y}}$ are well defined (if Dec queries both $\pi^{-1}(y)$ and $x^* \neq \pi^{-1}(y)$ we define its answer arbitrarily).

We now wish to consider the elements $x^* \in \{0,1\}^n$ for which $\mathrm{Breaker}^{\pi}(t, c) \neq \mathrm{Breaker}^{\pi_{|x^* \mapsto y}}(t, c)$. The set $\mathrm{Diff}^{\pi}(c, h, y)$ is a (possibly proper) superset of this set.

DEFINITION 3.3 (Diff)
For an oracle function Dec, a function $h \in \mathcal{H}_t$, and strings $c \in \{0,1\}^{\ell(t)}$ and $y \in \{0,1\}^n$, we let

$$\begin{aligned}\mathrm{Diff}^{\pi}(c, h, y) := \Big\{x^* \in \{0,1\}^n \ \Big|\ \exists k \in \{0,1\}^t \ : \ &\big(\mathrm{Dec}^{\pi}(k, c) \neq h(k) = \mathrm{Dec}^{\pi_{|x^* \mapsto y}}(k, c)\big) \\ &\vee \big(\mathrm{Dec}^{\pi}(k, c) = h(k) \neq \mathrm{Dec}^{\pi_{|x^* \mapsto y}}(k, c)\big)\Big\}.\end{aligned}$$

Note that for $x^* \notin \mathrm{Diff}^{\pi}(c, h_t, y)$ it holds that $\mathrm{Breaker}^{\pi}(t, c) = \mathrm{Breaker}^{\pi_{|x^* \mapsto y}}(t, c)$. To see this, let $k_0 \neq \perp$ be the lexicographic smaller output of the two calls. Clearly, k_0 must be the output of both calls to Breaker.

The next lemma states that if h is chosen from $\mathcal{H}_t$ (recall that this is a set of $(\ell(t) + t)$-wise independent hash-functions from $\{0,1\}^t$ to $\{0,1\}^{2t}$), then $\mathrm{Diff}^{\pi}(c, h, y)$ is very likely to be small for all c.

[7]The main difference between our approach and the one in [Sim98] is that we do not insist on keeping π a permutation. It turns out that this slackness makes our proof significantly simpler.

LEMMA 3.4
Let $y \in \{0,1\}^n$ and an oracle π be given. Assuming that $\mu_{\mathrm{Dec}}(t) < 2^t$, then

$$\Pr_{h \leftarrow \mathcal{H}_t}\left[\exists c \in \{0,1\}^{\ell(t)} \;:\; \left|\mathrm{Diff}^{\pi}(c,h,y)\right| \geq \mu_{\mathrm{Dec}}(t)^2\right] < 2^{-t} \;.$$

Proof. Fix c for now, and consider for every $k \in \{0,1\}^t$ the set $\mathcal{D}_{c,k}$ of all possible images of $\mathrm{Dec}^{\pi|_{x^*\mapsto y}}(k,c)$, enumerating over all $x^* \in \{0,1\}^n$, i.e., the set $\mathcal{D}_{c,k} := \left\{\mathrm{Dec}^{\pi|_{x^*\mapsto y}}(k,c) \;:\; x^* \in \{0,1\}^n\right\}$. (Note that the original image $\mathrm{Dec}^{\pi}(k,c)$ is also in this set.) We see that $|\mathcal{D}_{c,k}| \leq \mu_{\mathrm{Dec}}(t)+1 \leq 2^t$: in an execution of $\mathrm{Dec}^{\pi}(k,c)$ at most $\mu_{\mathrm{Dec}}(t)$ elements $x_1,\ldots,x_{\mu_{\mathrm{Dec}}(t)}$ elements are queried, and only if $x^* \in \{x_1,\ldots,x_{\mu_{\mathrm{Dec}}(t)}\}$ the image can change. By Lemma 2.3, (with $V = 2^t$, $s = t+\ell(t)$, $\alpha = \mu_{\mathrm{Dec}}(t)$, the B_k's are indicator variables for the event $h_t(k) \in \mathcal{D}_{c,k}$, and as mentioned before we assume $\mu_{\mathrm{Dec}}(t) \geq t+\ell(t)$), we get

$$\Pr_{h \leftarrow \mathcal{H}_t}\left[\left|\{k \in \{0,1\}^t \;:\; h(k) \in \mathcal{D}_{c,k}\}\right| > \mu_{\mathrm{Dec}}(t)\right] < \frac{t}{\mu_{\mathrm{Dec}}(t)^{-(t+\ell(t)-1)}} \leq 2^{-t-\ell(t)} \;. \tag{1}$$

Let $x^* \in \mathrm{Diff}^{\pi}(c,h,y)$, then there exists a $k \in \{0,1\}^t$ such that $\mathrm{Dec}^{\pi}(k,c) \neq h(k) = \mathrm{Dec}^{\pi|_{x^*\mapsto y}}(k,c)$ or $\mathrm{Dec}^{\pi}(k,c) = h(k) \neq \mathrm{Dec}^{\pi|_{x^*\mapsto y}}(k,c)$, and therefore, $h(k) \in \mathcal{D}_{c,k}$. For each such k, there are at most $\mu_{\mathrm{Dec}}(t)$ values $x^* \in \{0,1\}^n$ for which $\mathrm{Dec}^{\pi}(k,c) \neq \mathrm{Dec}^{\pi|_{x^*\mapsto y}}(k,c)$, giving a total of $\mu_{\mathrm{Dec}}(t)^2$ possible values. Thus, we get Lemma 3.4 by a union bound on all choices of $c \in \{0,1\}^{\ell}$. □

We next show that Lemma 3.4 yields that with high probability an adversary A behaves exactly the same given the oracle π or $\pi_{|x^*\mapsto y}$.

DEFINITION 3.5 (trace)
For a given oracle function Dec, the trace $\mathrm{tr}(\pi,y,r_A)$ of an adversary A is the sequence of all queries A makes to $\mathrm{Breaker}^{\pi}$ and π, and their responses, when it is run on input y and using random-coins r_A.

LEMMA 3.6
Let A be an adversary with oracle access to π and Breaker. Then, the following holds for every π and $y \in \{0,1\}^n$.

$$\Pr[\mathrm{tr}(\pi,y,r_A) \neq \mathrm{tr}(\pi_{|x^*\mapsto y},y,r_A)] < \mu_A(n)\left(2^{-\mu_{\mathrm{Dec}}^{-1}(n)} + \mu_{\mathrm{Dec}}(\mu_A(n))^2 2^{-n}\right) \;,$$

where the probability is over the choice of x^*, the randomness of A and the choice of the h_t done by Breaker.

Proof. Let $\mathcal{X}_\pi$ be all the queries to π in $\mathrm{tr}(\pi,y,r_A)$, clearly $|\mathcal{X}_\pi| \leq \mu_A(n)$. Let further $\mathcal{X}_{\mathrm{Diff}}$ be the union of the sets $\mathrm{Diff}^{\pi}(c,h_t,y)$ for all calls (c,t) made by A to Breaker. If $x^* \notin \mathcal{X}_\pi \cup \mathcal{X}_{\mathrm{Diff}}$, we have that $\mathrm{tr}(\pi,y,r_A) = \mathrm{tr}(\pi_{|x^*\mapsto y},y,r_A)$, since the behavior of Breaker will not change in such a call (cf. the remark after Definition 3.3). We Also note that for $t < \mu_{\mathrm{Dec}}^{-1}(n)$, it holds that $x^* \notin \mathrm{Diff}^{\pi}(c,h_t,y)$, as Dec cannot invoke π on such a long input. Thus, using Lemma 3.4 we have that with probability $1-\mu_A(n)2^{-\mu_{\mathrm{Dec}}^{-1}(n)}$ (over the randomness of Breaker)

$$\forall s \in \{1,\ldots,\mu_A(n)\} \;:\; |\,\mathrm{Diff}(c,h_t,y)| \leq \mu_{\mathrm{Dec}}(t)^2 \leq \mu_{\mathrm{Dec}}(\mu_A(n))^2 \tag{2}$$

If (2) holds, then $|\mathcal{X}_\pi \cup \mathcal{X}_{\mathrm{Diff}}| \leq \mu_A(n)\mu_{\mathrm{Dec}}(\mu_A(n))^2$. Hence, using union bound yields that the probability in question is at most $\mu_A(n)(2^{-\mu_{\mathrm{Dec}}^{-1}(n)} + \mu_{\mathrm{Dec}}(\mu_A(n))^2 2^{-n})$. □

Having the above we can finally prove Lemma 3.2.

Proof. (of Lemma 3.2) For a given value of π and $x_0,x_1 \in \{0,1\}^n$, let $\pi_{|x_0\leftrightarrow x_1} := \pi_{|x_0\mapsto\pi(x_1),x_1\mapsto\pi(x_0)}$. We assume without loss of generality that A queries $\pi(x)$ before it returns x. Then, the proof of the lemma easily follows by the next claim.

CLAIM 3.7
It holds that

$$\Pr_{\pi, x^* \leftarrow \{0,1\}^n, r_A}[\mathrm{tr}(\pi, y, r_A) \neq \mathrm{tr}(\pi_{|x^* \leftrightarrow \pi^{-1}(y)}, y, r_A)] < p(n) \ ,$$

where $p(n) := 2\mu_A(n)\left(2^{-\mu_{\mathrm{Dec}}^{-1}(n)} + \mu_{\mathrm{Dec}}(\mu_A(n))^2 2^{-n}\right)$.

Proof. If $\mathrm{tr}(\pi, y, r_A)$ and $\mathrm{tr}(\pi_{|x^* \leftrightarrow \pi^{-1}(y)}, y, r_A)$ are different, then one of $\mathrm{tr}(\pi, y, r_A) \neq \mathrm{tr}(\pi_{x^* \mapsto y}, y, r_A)$ or $\mathrm{tr}(\pi_{x^* \mapsto y}, y, r_A) \neq \mathrm{tr}(\pi_{|x^* \leftrightarrow \pi^{-1}(y)}, y, r_A)$ must hold. Since $\pi_{|x \leftrightarrow \pi^{-1}(y)}$ is also a permutation, and x^* is a uniformly chosen element given $\pi_{|x^* \leftrightarrow \pi^{-1}(y)}$ and y, Lemma 3.6 states that both these events have probability at most $\frac{p(n)}{2}$. □

The probability that $A^{(\pi, \mathrm{Breaker})}$ inverts y is at most the probability that the traces $\mathrm{tr}(\pi, y, r_A)$ and $\mathrm{tr}(\pi_{|x^* \leftrightarrow \pi^{-1}(y)}, y, r_A)$ are different plus 2^{-n} (to handle the case $x^* = \pi^{-1}(y)$). By the above claim, this is at most $2^{-n} + p(n)$, which concludes the proof. □

3.2 Obtaining the Impossibility Result

We first show that Breaker helps breaking the KDI-security of encryption schemes; for this purpose pairwise independent hash-functions are already sufficient.

LEMMA 3.8
Let $\mathcal{G}$ be a family of pairwise independent hash-functions from $\{0,1\}^t$ to $\{0,1\}^{2t}$. For any fixed π it holds that $\Pr_{k \leftarrow \{0,1\}^t, g \leftarrow \mathcal{G}}[\mathrm{Breaker}^\pi(n, \mathrm{Enc}^\pi(k, g(k)), g) = k] > 1 - \frac{1}{2^t}$.

Proof. By the pairwise independence of $\mathcal{G}$, once a key k and $g(k)$ is fixed, for $k' \neq k$ it holds that $g(k')$ is uniformly distributed over $\{0,1\}^{2t}$. Thus, the probability that a fixed $k' \neq k$ is returned by Breaker is $\frac{1}{2^{2t}}$. The union bound yields the lemma. □

With the above lemma, and the results from Section 3.1 we can now prove Theorem 1.1.

Proof. (of Theorem 1.1) Assume that $(\mathrm{Enc}, \mathrm{Dec})$ described above is KDI-secure against any $\mathrm{poly}(n)$-wise independent hash-function with black-box proof of security, and let A be the algorithm for inverting the one-way permutation guaranteed by this proof of security. Lemma 3.2 yields that the probability that A inverts y even given Breaker is at most $3\mu_A(n)(2^{-\mu_{\mathrm{Dec}}^{-1}(n)} + \mu_{\mathrm{Dec}}(\mu_A(n))^2 2^{-n})$, which is negligible since both A and Dec are efficient. On the other hand, Lemma 3.8 implies that Breaker can be used to win the KDI game with probability almost one. In particular, there exists a choice of π such that $(\mathrm{Enc}, \mathrm{Dec})$ is not KDI-secure for the family of $\mathrm{poly}(n)$-wise independent hash functions, yet π is not inverted by A. □

4 Arbitrary Assumptions

In this section we rule out the existence of reductions with strongly-black-box proof of security from KDI encryption to a very large class of hardness assumptions, i.e., we prove Theorem 1.2.

Let $(\mathrm{Enc}, \mathrm{Dec})$ be an encryption scheme, as in Section 3 we assume without loss of generality that the encryption scheme is always correct, has deterministic decryption algorithm, and is defined on messages of any polynomial length. We use the following notations. For a security parameter t (which for simplicity and using Proposition 2.9, we assume to be equal to the key length) let $\ell(t)$ be the length of an encryption of a message of length $2t$. We also use the following oracles. Let $\mathcal{H} = \left\{h_t : \{0,1\}^t \mapsto \{0,1\}^{2t}\right\}_{t \in \mathbb{N}}$ be an infinite sequence of functions. The next algorithm, Breaker, is as in Section 3, but uses slightly different parameters (e.g., it takes as input a description of a function, for example given as circuit).

ALGORITHM 4.1
The algorithm Breaker.

Inputs: An index $t \in \mathbb{N}$, a description of a function $f : \{0,1\}^t \times \{0,1\}^{\ell(t)} \mapsto \{0,1\}^{2t}$ and a ciphertext $c \in \{0,1\}^t$.

Description: Return the lexicographic smallest $k \in \{0,1\}^t$ with $f(k,c) = h_t(k)$, or $\perp$ if no such k exists.

In Lemma 4.3 we show that there is no KDI-secure cryptosystem in the presence of $(\mathrm{Breaker}, \mathcal{H})$, while in Lemma 4.4 we show that in many settings, having oracle access to Breaker does not yield any significant additional power. Finally, we use these lemmata to prove Theorem 1.2.

The following adversary uses $(\mathrm{Breaker}, \mathcal{H})$ to break the KDI-security of the encryption scheme.[8]

ALGORITHM 4.2
Algorithm $A_{\mathrm{KDI}}^{\mathrm{Breaker},\mathcal{H}}$.

Input: Security parameter t.

Step 1: Call $Q(h_t)$ (or $\widetilde{Q}(h_t)$) to obtain an encryption c of $h_t(k)$ (or of 0^{2t}).

Step 2: Call $\mathrm{Breaker}(t, \mathrm{Dec}, c)$ to obtain a candidate key k' or $\perp$.

Step 3: Output 1 iff Breaker did not return $\perp$.

LEMMA 4.3
$A_{\mathrm{KDI}}^{\mathrm{Breaker},\mathcal{H}}$ breaks the KDI security of the $(\mathrm{Enc}, \mathrm{Dec})$ with probability one over the choice of $\mathcal{H}$.

Proof. Algorithm A_{KDI} only gives the wrong answer if the oracle is $\widetilde{Q}$ and Breaker does not return $\perp$. In this case, the ciphertext c does not depend on h_t, and fixing it and a (correct) key k, the probability over the choice of h_t that $\widetilde{Q}$ does not return $\perp$ is (using the union bound) at most 2^{-t}. Therefore, using an averaging argument, the probability that h_t is such that something else but $\perp$ is returned with probability higher than $2^{-t/2}$ is at most $2^{-t/2}$.

Since the h_t's are chosen independently from each other, the probability that there exists $m_0 \in \mathbb{N}$ for which A_{KDI} does not break the scheme for some $t > t_0$, is zero. We conclude that with probability one over the choice of $h \in \mathcal{H}$, it holds that A_{KDI} breaks the KDI security of $(\mathrm{Enc}, \mathrm{Dec})$ infinitely often. $\square$

The next lemma shows that in the stand alone settings (unlike in the settings of Lemma 4.3) having oracle access to Breaker does not yield any significant additional power.

LEMMA 4.4
Let $A^{\mathrm{Breaker},h}$ be an algorithm with oracle access to $(\mathrm{Breaker}, h)$, and let $t_A(n)$, for security parameter n, be a polynomial-time computable upper bound on the running-time of $A^{\mathrm{Breaker},h}$.

Then for every polynomial computable function $\delta : \mathbb{N} \mapsto [0,1]$, there exists an algorithm $\widetilde{A}_\delta^h$, which has oracle access only to h, runs in time $\mathrm{poly}(\frac{1}{\delta(n)}, t_A(n))$ and uses random-coins of the same length as $A^{\mathrm{Breaker},h}$ such that the following holds. Conditioned that $A_\delta^{\mathrm{Breaker},h}$ and $\widetilde{A}_\delta^h$ are using the same random-coins, $A^{\mathrm{Breaker},h}(1^n) = \widetilde{A}_\delta^h(1^n)$ with probability $1 - \delta(n)$ over the choice of h.

[8] Algorithm A_{KDI} distinguishes whether the oracle is Q or $\widetilde{Q}$. One could also show that usually k' in the algorithm is the correct key.

Proof. Algorithm $\widetilde{A}$ emulates A, using its random-coins as the random-coins of A, while remembering all query and answer pairs to h. When the emulated A queries $\text{Breaker}(t, f, c)$, algorithm $\widetilde{A}$ distinguishes between the following two cases:

Case 1: $t < \log(t_A(n)) + \log(1/\delta(n))$. $\widetilde{A}$ fully emulates Breaker. Namely, $\widetilde{A}$ evaluates $h(k)$ for all $k \in \{0,1\}^t$ and returns the first one for which $f(k) = h(k)$. It returns $\perp$ if no such k exists.

Case 2: $t \geq \log(t_A(n)) + \log(1/\delta(n))$. $\widetilde{A}$ checks all the previous queries to h of length t in lexicographic order. If for one of those queries it holds that $f(k) = h(k)$, it returns k, otherwise it returns $\perp$.

The bound on the running-time of $\widetilde{A}$ is clear. Note that in the second case, the probability (over the choice of h) that $\widetilde{A}$ uses a different value in the simulation from what Breaker would return is at most $2^{-\log(t_A(n))-\log(1/\delta(n))} = \frac{\delta(n)}{t_A(n)}$ (in the first case, we always return the correct value). Since there are at most $t_A(n)$ calls to Breaker, the probability that in any of those $\widetilde{A}$ returns a wrong value is at most $\delta(n)$, which proves the lemma. $\square$

We now combine these lemmata to prove Theorem 1.2.

Proof. (of Theorem 1.2) Assume that there exists a strongly-black-box proof of security from (Enc, Dec) to a cryptographic game Γ and let $M^{(\cdot)}$ be the algorithm for breaking Γ as guaranteed by this proof of security. Lemma 4.3 yields that $A_{\text{KDI}}^{\text{Breaker},h}$ breaks (Enc, Dec) with probability one over the choice of h. Thus, $M^{A_{KDI}^{\text{Breaker},h}}$ breaks the security of Γ with probability one over the choice of h. Namely,

$$\mathsf{Pr}_h\left[\mathsf{Pr}_{r_A,r_\Gamma}[M^{A_{KDI}^{\text{Breaker},h}} \leftrightarrow \Gamma(1^n) \text{ wins}] > \frac{1}{p_h(n)} \qquad \text{for infinitely many } n\right] = 1 \ , \tag{3}$$

where r_A and r_Γ denote the random coins of A and Γ, respectively, and p_h is some polynomial that may depend on h. In the following we first remove the dependence of the polynomial p_h from h. For this let

$$\varepsilon(n) := \mathsf{Pr}_{h,r_A,r_\Gamma}[M^{A_{KDI}^{\text{Breaker},h}} \leftrightarrow \Gamma(1^n) \text{ wins}] = \mathsf{E}_h\left[\mathsf{Pr}_{r_A,r_\Gamma}[M^{A_{KDI}^{\text{Breaker},h}} \leftrightarrow \Gamma(1^n) \text{ wins}]\right] \ .$$

We show that ε is non-negligible. Using Markov bound we get for every $n \in \mathbb{N}$ that

$$\mathsf{Pr}_h\left[\mathsf{Pr}_{r_A,r_\Gamma}[M^{A_{KDI}^{\text{Breaker},h}} \leftrightarrow \Gamma(1^n) \text{ wins}] < n^2\varepsilon(n)\right] > 1 - \frac{1}{n^2} \ ,$$

and therefore[9]

$$\mathsf{Pr}_h\left[\mathsf{Pr}_{r_A,r_\Gamma}[M^{A_{KDI}^{\text{Breaker},h}} \leftrightarrow \Gamma(1^n) \text{ wins}] < n^2\varepsilon(n) \qquad \text{for all } n > 2\right] > 1 - \sum_{n=2}^{\infty}\frac{1}{n^2} > \frac{1}{3} \tag{4}$$

Combining (3) and (4) we get

$$\mathsf{Pr}_h\left[\frac{1}{p_h(n)} < \mathsf{Pr}_{r_A,r_\Gamma}[M^{A_{KDI}^{\text{Breaker},h}} \leftrightarrow \Gamma(1^n) \text{ wins}] < n^2\varepsilon(n) \qquad \text{for infinitely many } n\right] > \frac{1}{3} \ , \tag{5}$$

which implies that there is a polynomial $p(n)$ such that $\varepsilon(n) > \frac{1}{p(n)}$ infinitely often.

We would like now to apply Lemma 4.4 on $M^{A_{KDI}^{\text{Breaker},h}}$. Recall that Lemma 4.4 was proved only in the stand alone settings, where in particular no interaction with a random system is considered. Since the proof of security of (Enc, Dec) is strongly-black-box, however, we have that Γ, through interaction with $M^{A_{KDI}^{\text{Breaker},h}}$, does not access h. Therefore, Γ's answers are determined by the output behavior of $M^{A_{KDI}^{\text{Breaker},h}}$ and the proof of Lemma 4.4 goes through also in this setting. Hence, Lemma 4.4 yields, letting $\delta(n) = \frac{1}{2p(n)}$, the existence

[9]The σ-additivity of the measure implies that the event in the next probability is measurable.

an efficient algorithm $\widetilde{M}^h$ with oracle access *only* to h, such that $\mathsf{Pr}_{h,r_A,r_\Gamma}[\widetilde{M}^h \leftrightarrow \Gamma(1^n) \text{ wins}] > \frac{1}{2p(n)}$ for infinitely many n's.

Our final step is to emulate $\widetilde{M}^h$, where rather than accessing h we randomly chooses the answer of each time one is requested (and cache it). The latter emulation breaks the cryptographic assumption with probability at least $\frac{1}{2p(n)}$ for infinitely many n's and since it is also efficient, it implies that Γ is not secure. $\square$

5 Applying Our Technique to Other Primitives

It seems tempting to try and use the above Breaker also to show the impossibility of constructing other KDI-secure primitives. Consider for instance pseudorandom functions or permutations which are supposed to be secure even if the adversary can obtain its value on a function of its secret key. Halevi and Krawczyk [HK07] show that a deterministic construction cannot exist, but give a construction in case the permutation has an additional public parameter (i.e., *salt*) chosen after the challenging function is fixed. Their construction, however, compresses (e.g., maps n bits to $n/2$).

It is indeed possible to generalize our techniques to this case, as long for as the pseudorandom functions are injective for *every key*. In this case, Breaker finds a key k such that $f^{k,r}(h(k)) = c$, where f is the pseudorandom function and r is the random salt. The reason this method fails if the construction compresses (as the one given by Halevi and Krawczyk), is that Breaker as defined above does not seem to give useful information about the key anymore, since it is unlikely that the correct key is the lexicographically smallest.

It seems that we cannot utilize our Breaker also for the general case of length increasing pseudorandom functions (or equivalently, for the case that we are allowed to make several KDI queries). Consider the question whether a given pseudorandom function is constant on a negligible fraction of the keys (e.g., on a single key k it holds that $f^{k,r}(\cdot) := 0^\ell$). Deciding whether a given function has this property or not might be infeasible. Yet, using for instance the Breaker of Section 4, we can easily find the right answer: ask the Breaker on $(h, 0^\ell)$, where h is a random hash function, and answer "Yes" is the Breaker finds some consistent key. Thus, in this setting our Breaker gives us an extra power that we cannot emulate.

Acknowledgment

We are very grateful to Oded Goldreich, Jonathan Hoch, Gil Segev and Omer Reingold for useful discussion.

References

[AR02] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 15(2):103–127, 2002.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75, 2002.

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

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

[DY83] Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.

[GGKT05] Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM Journal on Computing*, 35(1):217–246, 2005.

[GKM$^+$00] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, 2000.

[Gol01] Oded Goldreich. Randomized methods in computation - lecture notes. 2001.

[GT00] Rosario Gennaro and Luca Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, 2000.

[HHRS07] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols – A tight lower bound on the round complexity of statistically-hiding commitments. In *Proceedings of the 47th Annual Symposium on Foundations of Computer Science (FOCS)*, 2007.

[HK05] Omer Horvitz and Jonathan Katz. Bounds on the efficiency of "black-box" commitment schemes. In *ICALP '05*, pages 128–139, 2005.

[HK07] Shai Halevi and Hugo Krawczyk. Security under key-dependent inputs. In Sabrina De Capitani di Vimercati and Paul Syverson, editors, *14th ACM conference on Computer and communications security*, pages 466–475, 2007.

[HU08] Dennis Hofheinz and Dominique Unruh. Towards key-dependent message security in the standard model. In Nigel Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, 2008.

[IJK07] Russell Impagliazzo, Ragesh Jaiswal, and Valentine Kabanets. Chernoff-type direct product theorems. In *Advances in Cryptology – CRYPTO 2007*, pages 500–516, 2007.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 44–61, 1989.

[KST99] Jeong Han Kim, Daniel R. Simon, and Prasad Tetali. Limits on the efficiency of one-way permutation-based hash functions. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 535–542, 1999.

[RTV04a] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20, 2004.

[RTV04b] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2004.

[Rud88] Steven Rudich. *Limits on the Provable Consequences of One-Way Functions.* PhD thesis, U.C. Berkeley, 1988.

[Sim98] Daniel Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology – EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345. Springer, 1998.

[Wee07] Hoeteck Wee. One-way permutations, interactive hashing and statistically hiding commitments. In *Theory of Cryptography, Fourth Theory of Cryptography Conference, TCC 2007*, pages 419–433, 2007.

A Gennaro-Trevisan Style Proof

In this section, we prove a lemma which is analogous to Lemma 3.2 using the technique introduced by Gennaro and Trevisan [GT00]. For this purpose we use Lemma 3.4. Since the bound holds against non-uniform algorithms as well, we can assume that A is deterministic.

LEMMA A.1

Fix a deterministic algorithm A with oracle access to π and Breaker, and assume that

$$\Pr_{y\leftarrow\{0,1\}^n}[A^{(\pi,\mathrm{Breaker})}(y)=\pi^{-1}(y)] > \varepsilon(n)+\mu_A(n)\cdot 2^{-\mu_{\mathrm{Dec}}^{-1}(n)} \ .$$

Then, π can be described using $\log((2^n-s(n))!)+2s(n)\log\Big(\frac{e2^n}{s(n)}\Big)+\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))$ bits, where $s(n)=\frac{\varepsilon(n)2^n}{4\mu_A(n)\cdot\mu_{\mathrm{Dec}}(\mu_A(n))^2}$.

For example, assume that $\mu_{\mathrm{Dec}}(n)\le n^5$, $\mu_A(n)\le 2^{n/30}$ and $\varepsilon(n)=\frac{1}{\mu_A(n)}$. Lemma A.1 yields that the fraction of oracles that A can invert with probability $\varepsilon(n)+\mu_A(n)\cdot 2^{-n^{1/5}}<2\varepsilon(n)$ is at most

$$\frac{(2^n-s(n))!\big(\frac{e2^n}{s(n)}\big)^{s(n)}2^{\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))}}{(2^n)!}\le\frac{\big(\frac{e2^n}{s(n)}\big)^{s(n)}2^{\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))}}{(2^n-s(n))^{s(n)}}$$
$$=\Big(\frac{e2^n}{s(n)\underbrace{(2^n-s(n))}_{\ge 2^{n-1}}}\Big)^{s(n)}2^{\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))}\le\Big(\frac{2e}{s(n)}\Big)^{s(n)}2^{\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))}\le 2^{-2^{n/2}} \ .$$

Proof. We assume $\varepsilon(n)>2\mu_A(n)2^{-n}$, which is without loss of generality, since otherwise the statement is trivial.

Our description of π consists of the following parts: the description a set $S\subseteq\{0,1\}^n$, the description of the image of S under π (which roughly corresponds to the y's on which A succeeds in inverting), and the description of the permutation which π implies if restricted on $\{0,1\}^n\setminus S$, i.e., the elements not in S. Also, we will save a fixing of Breaker's random coins (i.e., the choice of the functions h_m for $m\le\mu_A(n)$).

The description of S and the image of S both require $\log\big(\binom{2^n}{|S|}\big)\le|S|\log(e\frac{2^n}{|S|})$ bits. The description of the permutation requires at most $\log((2^n-|S|)!)$ bits. To store the functions h_m takes $\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))$ bits, for some appropriate family $\mathcal{H}$. Thus, in total $\log((2^n-|S|)!)+2|S|\log(e\frac{2^n}{|S|})+\mu_A(n)^2\mu_{\mathrm{Dec}}(\mu_A(n))$ bits are sufficient. In the following we succeed in making S as big as $|S|=\frac{\varepsilon(n)2^n}{4\mu_A(n)\cdot\mu_{\mathrm{Dec}}(\mu_A(n))^2}$, which implies our upper bound on the description size of π.

Fix now an oracle π such that $\Pr_{y\leftarrow\{0,1\}^n}[A^{(\pi,\mathrm{Breaker})}(y)=\pi^{-1}(y)]>\varepsilon(n)+\mu_A(n)2^{-\mu_{\mathrm{Dec}}^{-1}(n)}$. Let $\mathrm{Bad}(y)$ be the event that $A(y)$ makes a query $\mathrm{Breaker}^\pi(t,c)$, with $|\mathrm{Diff}^\pi(c,h_t,y)|>\mu_{\mathrm{Dec}}(t)^2$, where h_t is the function chosen by Breaker for length t. By Lemma 3.4 and our assumption about the success probability of A, we have that $\Pr_{y\leftarrow\{0,1\}^n}[A^{(\pi,\mathrm{Breaker})}(y)=\pi^{-1}(y)\wedge\overline{\mathrm{Bad}(y)}]>\varepsilon(n)$. Since Breaker selects the hash functions it returns at random, a simple average argument yields that there exists a fixing r of the randomness used by Breaker to chose the h_t such that $\Big|I:=\{y\in\{0,1\}^n\ :\ A(y)=\pi^{-1}(y)\}\wedge\overline{\mathrm{Bad}(y)}\Big|>\varepsilon(n)2^{n-1}$. We store this r as part of the description of π and denote by $\mathrm{Breaker}_r$ the oracle Breaker with this fixing.

The set S. We use the following, inefficient, algorithm to create S: first, remove the lexicographic smallest element y^* from I and add $\pi^{-1}(y^*)$ to S. Simulate $A^{(\pi,\mathrm{Breaker}_r)}(y^*)$, and remove all queries A makes whose answers are in I from I (without putting them into S). For queries to $\mathrm{Breaker}_r^\pi(c,h)$, consider all $x^*\in\mathrm{Diff}^\pi(c,h,y)$ and remove their images from I as well (y itself cannot be removed from I, as we already removed it). Once the simulation is finished, repeat with the next element. Clearly, for every simulation we remove at most $\mu_A(n) + \mu_A(n) \cdot \mu_{\text{Dec}}(\mu_A(n))^2$ elements from I before moving another element to S, which ensures $|S| \geq \frac{\varepsilon(n)2^n}{4\mu_A(n)\cdot\mu_{\text{Dec}}(\mu_A(n))^2}$.

Reconstruction π. We now show that we can reconstruct π from the given information. For this, we first reconstruct the oracle outside of S from the given information. Then pick the lexicographic smallest element y^* whose preimage is not yet known, and simulate $A^{\pi,\text{Breaker}_r}(y^*)$. If A calls $\pi(x)$ for some x whose image is not known yet, then the image must be y (as otherwise we would have removed the element from I and stored the value of $\pi(x)$ explicitly). If A calls $\text{Breaker}_r^\pi(c,h)$, note that if k is returned, all queries to $\text{Dec}^\pi(k,c)$ but the one of $\pi^{-1}(y)$ (if it occurs) are also known. We try all candidates x^* (i.e., the elements whose image we don't know at this point) for $\pi^{-1}(y^*)$, and simulate $\text{Dec}^{\pi|_{x^*\to y^*}}(k,c)$, which we can do unless a query is made whose answer we don't know, in which case it must be a wrong k or x^*. Now, if one simulation satisfies $\text{Dec}^{\pi|_{x^*\to y^*}}(k,c) = h(k)$, then k must be the correct answer to this Breaker_r call, since otherwise we would have removed all the queries from I. Therefore, we can simulate A's run correctly and obtain the correct $\pi^{-1}(y^*)$ as output of A. □