Chosen-Ciphertext Security via Correlated Products[*]

Alon Rosen[†] Gil Segev[‡]

Abstract

We initiate the study of one-wayness under *correlated products*. We are interested in identifying necessary and sufficient conditions for a function f and a distribution on inputs $(x_1, \ldots, x_k)$, so that the function $(f(x_1), \ldots, f(x_k))$ is one-way. The main motivation of this study is the construction of public-key encryption schemes that are secure against chosen-ciphertext attacks (CCA). We show that any collection of injective trapdoor functions that is secure under a very natural correlated product can be used to construct a CCA-secure public-key encryption scheme. The construction is simple, black-box, and admits a direct proof of security. It can be viewed as a simplification of the seminal work of Dolev, Dwork and Naor (SICOMP '00), while relying on a seemingly incomparable assumption.

We provide evidence that security under correlated products is achievable by demonstrating that lossy trapdoor functions (Peikert and Waters, STOC '08) yield injective trapdoor functions that are secure under the above mentioned correlated product. Although we currently base security under correlated products on existing constructions of lossy trapdoor functions, we argue that the former notion is potentially weaker as a general assumption. Specifically, there is no fully-black-box construction of lossy trapdoor functions from trapdoor functions that are secure under correlated products.

[*]A preliminary version of this work appeared in *Proceedings of the 6th Theory of Cryptography Conference (TCC)*, pages 419–436, 2009.

[†]Efi Arazi School of Computer Science, Herzliya Interdisciplinary Center (IDC), Herzliya 46150, Israel. Email: `alon.rosen@idc.ac.il`. Research supported in part by BSF grant 2006317.

[‡]Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: `gil.segev@weizmann.ac.il`. Research supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

1 Introduction

The construction of secure public-key encryption schemes lies at the heart of cryptography. Following the seminal work of Goldwasser and Micali [26], increasingly strong security definitions have been formulated. The strongest notion to date is that of semantic security against a chosen-ciphertext attack (CCA) [42, 47], which protects against an adversary that is given access to decryptions of ciphertexts of her choice.

Constructions of CCA-secure public-key encryption schemes have followed several structural approaches. These approaches, however, either result in rather complicated schemes, or rely only on specific number-theoretic assumptions. Our goal in this paper is to construct a simple CCA-secure public-key encryption scheme based on general computational assumptions.

The first approach for constructing a CCA-secure public-key encryption scheme was put forward by Naor and Yung [42], and relies on any semantically secure public-key encryption scheme and non-interactive zero-knowledge (NIZK) proof system for $\mathcal{NP}$. Their approach was later extended by Dolev, Dwork and Naor [13] for a more general notion of chosen-ciphertext attack, and subsequently simplified by Sahai [50] and by Lindell [37]. Encryption schemes resulting from this approach, however, are somewhat complicated and impractical due to the use of generic NIZK proofs.

An additional approach was introduced by Cramer and Shoup [12], and is based on "hash proof systems", which were shown to exist based on several number-theoretic assumptions (see also the refinement by Kiltz et al. [35] and the references therein). Elkind and Sahai [15] observed that both the above approaches can be viewed as special cases of a single paradigm in which ciphertexts include "proofs of well-formedness". Even though in some cases this paradigm leads to elegant and efficient constructions [11], the complexity of the underlying notions makes the general framework somewhat cumbersome.

Recently, Peikert and Waters [46] introduced the intriguing notion of lossy trapdoor functions, and demonstrated that such functions can be used to construct a CCA-secure public-key encryption scheme in a black-box manner. Their construction can be viewed as an efficient and elegant realization of the "proofs of well-formedness" paradigm. Lossy trapdoor functions seem to be a very powerful primitive. In particular, they were shown to also imply oblivious transfer protocols and collision-resistant hash functions.[1] It is thus conceivable that CCA-secure encryption can be realized based on weaker primitives.

A different approach was suggested by Canetti, Halevi and Katz [6] (followed by [3, 4, 5]) who constructed a CCA-secure public-key encryption scheme based on any identity-based encryption (IBE) scheme. Their construction is elegant, black-box, and essentially preserves the efficiency of the underlying IBE scheme. However, IBE is a rather strong cryptographic primitive, which is currently realized only based on a small number of specific number-theoretic assumptions.

Finally, a recent line of research has focused on basing chosen-ciphertext security on *computational* number-theoretic problems, as opposed to *decisional* number-theoretic problems [7, 10, 29, 32]. Most notably, Cash, Kiltz, and Shoup [7] proposed the first efficient CCA-secure scheme based on the computational Diffie-Hellman assumption, and Hofheinz and Kiltz [32] proposed the first efficient CCA-secure encryption scheme based on the factoring assumption.

[1] We note that the constructions of CCA-secure encryption and collision-resistant hash functions presented in [46] require lossy trapdoor functions that are "sufficiently lossy" (i.e., the constructions rely on lossy trapdoor functions with sufficiently good parameters).

1.1 Our Contributions

Motivated by the task of constructing a simple CCA-secure public-key encryption scheme, we initiate the study of one-wayness under *correlated products*. The main question in this context is to identify necessary and sufficient conditions for a collection of functions $\mathcal{F}$ and a distribution on inputs $(x_1, \dots, x_k)$ so that the function $(f_1(x_1), \dots, f_k(x_k))$ is one-way, where $f_1, \dots, f_k$ are independently chosen from $\mathcal{F}$. Our results are as follows:

1. We show that any collection of injective trapdoor functions that is secure under a very natural correlated product can be used to construct a CCA-secure public-key encryption scheme. The construction is simple, black-box, and admits a direct proof of security. Arguably, both the underlying assumption and the proof of security are simple enough to be taught in an undergraduate course in cryptography.

2. We demonstrate that any collection of lossy trapdoor functions (with appropriately chosen parameters) yields a collection of injective trapdoor functions that is secure under the correlated product that is required by our encryption scheme. In turn, existing constructions of lossy trapdoor functions [1, 46, 49] imply that our encryption scheme can be based on the hardness of the decisional Diffie-Hellman problem, and of Paillier's decisional composite residuosity problem.

3. We argue that security under correlated products is potentially weaker than lossy trapdoor functions as a general computational assumption. Specifically, we prove that there is no fully-black-box construction of lossy trapdoor functions from trapdoor functions (and even from enhanced trapdoor permutations) that are secure under correlated products.

In the remainder of this section we provide a high-level overview of our contributions, and then turn to describe the related and subsequent work.

1.2 Security Under Correlated Products

It is well known that for every collection of one-way functions $\mathcal{F} = \{f_s\}_{s \in S}$ and polynomially-bounded $k \in \mathbb{N}$, the collection $\mathcal{F}_k = \{f_{s_1,\dots,s_k}\}_{(s_1,\dots,s_k) \in S^k}$, whose members are defined as

$$f_{s_1,\dots,s_k}(x_1, \dots, x_k) = (f_{s_1}(x_1), \dots, f_{s_k}(x_k))$$

is also one-way. Moreover, such a direct product amplifies the one-wayness of $\mathcal{F}$ [24, 53], and this holds even when considering a single function (i.e., when $s_1 = \cdots = s_k$).

In general, however, the one-wayness of $\mathcal{F}_k$ is guaranteed only when the inputs are independently chosen, and when the inputs are correlated no such guarantee can exist. A well-known example for insecurity under correlated products is Håstad's attack [2, 30] on plain-broadcast RSA: there is an efficient algorithm that is given as input $x^3 \bmod N_1$, $x^3 \bmod N_2$, and $x^3 \bmod N_3$, and outputs x. More generally, it is rather easy to show that if collections of one-way functions exist, then there exists a collection of one-way functions $\mathcal{F} = \{f_s\}_{s \in S}$ such that $f_{s_1,s_2}(x, x) = (f_{s_1}(x), f_{s_2}(x))$ is not one-way. However, this does not rule out the possibility of constructing a collection of one-way functions whose product remains one-way even when the inputs are correlated.

Informally, given a collection $\mathcal{F}$ of functions and a distribution $\mathcal{C}_k$ of inputs $(x_1, \dots, x_k)$, we say that $\mathcal{F}$ is *secure under a $\mathcal{C}_k$-correlated product* if $\mathcal{F}_k$ is one-way when the inputs $(x_1, \dots, x_k)$ are distributed according to $\mathcal{C}_k$ (a formal definition is provided in Section 3). The main goal in this setting is to characterize the class of collections $\mathcal{F}$ and distributions $\mathcal{C}_k$ that satisfy this notion.

The study of cryptographic primitives that retain their security even when their inputs are correlated has already been considered before. Specifically, Naor and Reingold [41] put forward the notion of pseudorandom synthesizers, whose outputs are required to be computationally indistinguishable from random even if their inputs are correlated. We note that one-wayness under correlated inputs differs from the notion of synthesizers in that the latter refer to pseudorandomness, rather than one-wayness.

We motivate the study of security under correlated products by relating it to the study of chosen-ciphertext security. Specifically, we show that any collection of injective trapdoor functions that is secure under a very natural correlated product can be used to construct a CCA-secure public-key encryption scheme. The simplest form of distribution $\mathcal{C}_k$ on inputs that is sufficient for our construction is the *uniform k-repetition distribution* that outputs k copies of a uniformly chosen input x. We note that although this seems to be a strong requirement, we demonstrate that it can be based on various number-theoretic assumptions.

More generally, our construction can rely on any distribution $\mathcal{C}_k$ with the property that any $(x_1, \dots, x_k)$ in the support of $\mathcal{C}_k$ can be reconstructed given any $t = (1-\epsilon)k$ entries from $(x_1, \dots, x_k)$, for some constant $0 < \epsilon < 1$. For example, $\mathcal{C}_k$ may be a distribution that evaluates a random polynomial of degree at most $t-1$ on a fixed set of k points (in this case the values x_i's are t-wise independent, but other choices which do not guarantee such a strong property are also possible).

1.3 Chosen-Ciphertext Security via Correlated Products

Consider the following, very simple, public-key encryption scheme. The public-key consists of an injective trapdoor function f, and the secret-key consists of its trapdoor td. Given a message $m \in \{0,1\}$, the encryption algorithm chooses a random input x and outputs the ciphertext $(f(x), m \oplus h(x))$, where h is a hard-core predicate of f. The decryption algorithm uses the trapdoor to retrieve x and then extracts m. In what follows we frame our approach as a generalization of this fundamental scheme.

The above scheme is easily proven secure against a chosen-plaintext attack. Any adversary $\mathcal{A}$ that distinguishes between an encryption of 0 and an encryption of 1 can be used to construct an adversary $\mathcal{A}'$ that distinguishes between $h(x)$ and a randomly chosen bit with exactly the same probability. Specifically, $\mathcal{A}'$ receives a function f, a value $y = f(x)$, and a bit w (which is either $h(x)$ or a uniformly chosen bit), and emulates $\mathcal{A}$ with f as the public-key and $(y, m \oplus w)$ as the challenge ciphertext for a random message m. This scheme, however, fails to be proven secure against a chosen-ciphertext attack (even when considering only CCA1 security). There is a conflict between the fact that $\mathcal{A}'$ is required to answer decryption queries, and the fact that $\mathcal{A}'$ does not have the trapdoor for inverting f.

The following simplified variant of our scheme is designed to resolve this conflict. The public-key consists of k pairs of functions $(f_1^0, f_1^1), \dots, (f_k^0, f_k^1)$, where each function is sampled independently from a collection $\mathcal{F}$ of injective trapdoor functions.[2] The secret-key consists of the trapdoors $(td_1^0, td_1^1), \dots, (td_k^0, td_k^1)$, where each td_i^b is the trapdoor of the function f_i^b. Given a message $m \in \{0,1\}$, the encryption algorithm chooses a random $v = v_1 \cdots v_k \in \{0,1\}^k$, a random input x, and outputs the ciphertext

$$E_{PK}(m; v, x) = (v, f_1^{v_1}(x), \dots, f_k^{v_k}(x), m \oplus h(x)) \ ,$$

where h is a hard-core predicate of $\mathcal{F}_k$ with respect to the uniform k-repetition distribution. The

[2] For CCA1 security any $k = \omega(\log n)$ is sufficient, where n is the security parameter. For our more generalized construction that guarantees CCA2 security, any $k = n^\epsilon$ for some constant $0 < \epsilon < 1$ is sufficient.

decryption algorithm acts as follows: given a ciphertext $(v, y_1, \dots, y_k, z)$ it inverts $y_1, \dots, y_k$ to obtain $x_1, \dots, x_k$, and if $x_1 = \cdots = x_k$ then it outputs $h(x_1) \oplus z$ (otherwise it outputs $\perp$).

In order to prove the CCA1 security of this scheme, we show that any adversary $\mathcal{A}$ that breaks the CCA1 security of the scheme can be used to construct an adversary $\mathcal{A}'$ that distinguishes between $h(x)$ and a randomly chosen bit with exactly the same probability. The adversary $\mathcal{A}'$ receives as input k functions $f_1, \dots, f_k \in \mathcal{F}$, k values $y_1 = f_1(x), \dots, y_k = f_k(x)$, and a bit w (which is either $h(x)$ or a uniformly chosen bit). $\mathcal{A}'$ simulates the CCA1 interaction to $\mathcal{A}$ by choosing a random value $v^* = v_1^* \cdots v_k^* \in \{0,1\}^k$, and for each pair (f_i^0, f_i^1) it sets $f_i^{v_i^*} = f_i$ and samples $f_i^{1-v_i^*}$ together with its trapdoor from $\mathcal{F}$. Note that now $\mathcal{A}'$ is able to answer decryption queries as long as none of them contain the value v^*, and in this case we claim that essentially no information on v^* is revealed. The challenge ciphertext is then computed as $(v^*, y_1, \dots, y_k, m \oplus w)$ for a random message m. If $\mathcal{A}$ guesses the bit m correctly then $\mathcal{A}'$ outputs that $w = h(x)$, and otherwise $\mathcal{A}'$ outputs that w is a random bit.

Our scheme can be viewed as a realization of the Naor-Yung paradigm [42, 13] in which a message is encrypted using several independently chosen keys, and ciphertexts include "proofs of well-formedness". Specifically, whereas our public key consists of descriptions of *functions*, the public key in the Naor-Yung paradigm consists of *public keys* for a semantically-secure encryption scheme, together with a reference string for an NIZK proof system. A message is then encrypted using several keys, and also includes an NIZK proof that all ciphertexts are indeed encryptions of the same message. Our scheme simplifies their approach by allowing the decryption algorithm to verify "well-formedness" of ciphertexts without any additional "proof": given any one of the trapdoors it is possible to verify that the remaining values are consistent with the same input x by simply evaluating the remaining functions on the input x. This is the advantage of using functions and not (randomized) encryption schemes. A disadvantage of our approach when compared to the Naor-Yung paradigm, is that even for achieving only CCA1 security we need a super-logarithmic number of functions, whereas in the scheme of Naor and Yung [42] it suffices to use only two public keys (and an NIZK proof system). The reason is that in our approach the value v^* in the challenge ciphertext must be unpredictable to an adversary, and in [42] the value v^* actually does not exist since the verification of well-formedness is done using the NIZK proof system.

In addition, we note that our underlying assumption is incomparable to that required by the scheme of Dolev, Dwork and Naor [13]: although we require security under correlated products, we can rely on injective trapdoor functions, whereas their scheme (currently) requires enhanced trapdoor permutations for constructing the NIZK proof system.

Our scheme is inspired also by the one based on lossy trapdoor functions [46], and specifically, by the generic construction of *all-but-one* lossy trapdoor functions from lossy trapdoor functions. However, the proof security of our construction is simpler than that of [46] due to the additional hybrids resulting from using both lossy trapdoor functions and all-but-one trapdoor functions. In addition, our construction only relies on *computational* hardness, whereas the construction of [46] relies on the *statistical* properties of lossy trapdoor functions.

Finally, we note that our proof of security is rather similar to that of the IBE-based schemes [4, 5, 6]. The value v^* can be viewed as the challenge identity, for which $\mathcal{A}'$ does not have the secret key, and is therefore not able to decrypt ciphertexts for this identity. For any other identity $v \neq v^*$, $\mathcal{A}'$ has sufficient information to decrypt ciphertexts.

In some sense, our approach enjoys "the best of both worlds" in that both the underlying assumption and the proof of security are simpler than those of previous approaches.

1.4 A Black-Box Separation

Although we currently base security under correlated products on lossy trapdoor functions, we argue that security under correlated products is potentially weaker than lossy trapdoor functions as a general computational assumption. Specifically, we prove that there is no fully-black-box construction of lossy trapdoor functions from trapdoor functions that are secure under correlated products. Moreover, this holds also for lossy functions (i.e., the same as lossy trapdoor functions but without a trapdoor). We present an oracle relative to which there exists a collection of injective trapdoor functions (and even of enhanced trapdoor permutations) that is secure under a correlated product with respect to the above mentioned uniform k-repetition distribution, but there is no collection of lossy trapdoor functions. The oracle is essentially the collision-finding oracle due to Simon [51], and the proof follows the approach of Haitner et al. [28] while overcoming several technical difficulties.

Informally, consider a circuit A which is given as input $(f_1(x), \ldots, f_k(x))$, and whose goal is to retrieve x. The circuit A is provided access to an oracle Sam that receives as input a circuit C and outputs random w and w' such that $C(w) = C(w')$. As in the approach of Haitner et al. the idea underlying the proof is to distinguish between two cases: one in which A obtains information on x via one of its Sam-queries, and the other in which none of A's Sam-queries provides information on x. The proof consists of two modular parts dealing with these two cases separately. In the first part we generalize an argument of Haitner et al. (who in turn generalized the reconstruction lemma of Gennaro and Trevisan [18]) to deal with the product of several functions. We show that the probability that $\mathcal{A}$ retrieves x in the first case is exponentially small. In the second part we show that the second case can essentially be reduced to the first case. This part of the proof is simpler than the corresponding argument of Haitner et al. that considers a more interactive setting.

1.5 Related and Subsequent Work

Related work. Much research has been devoted for the construction of CCA-secure public-key encryption schemes. A significant part of this research was already mentioned in the previous sections, and here we mainly focus on recent results regarding the possibility and limitations of basing such schemes on general computational assumptions.

Pass, shelat and Vaikuntanathan [43] constructed a public-key encryption scheme that is non-malleable against a chosen-plaintext attack from any semantically secure one (building on the scheme of Dolev, Dwork and Naor [13]). Their technique was later shown by Cramer et al. [9] to also imply non-malleability against a weak notion of chosen-ciphertext attack, in which the number of decryption queries is bounded. These approaches, however, are rather impractical due to the use of designated verifier NIZK proofs that are constructed somewhat inefficiently from any public-key encryption scheme. Choi et al. [8] then showed that the latter notions of security can in fact be elegantly realized in a black-box manner based on the same assumptions. The reader is referred to [13, 44] for classifications of the different notions of security.

Impagliazzo and Rudich [34] introduced a paradigm for proving impossibility results for cryptographic constructions. They showed that there are no black-box constructions of key-agreement protocols from one-way permutations, and substantial additional work in this line followed (see, for example [17, 19, 21, 36, 51] and many more). The reader is referred to [48] for a comprehensive discussion and taxonomy of black-box constructions. In the context of public-key encryption schemes, most relevant to our result is the work of Gertner, Malkin and Myers [20], who addressed the question of whether or not semantically secure public-key encryption schemes imply the existence of CCA-secure schemes. They showed that there are no black-box constructions in which the decryption algorithm of the proposed CCA-secure scheme does not query the encryption algorithm of the semantically secure one.

Subsequent work. Following our work, Peikert [45] and Goldwasser and Vaikuntanathan [27] showed that security under correlated products is achievable also under the worst-case hardness of lattice problems (although these assumptions are currently not known to imply lossy trapdoor functions with the appropriately chosen parameters that are required for our transformation). Both Peikert, and Goldwasser and Vaikuntanathan, show that our framework for chosen-ciphertext security can be extended to deal with a relaxed form of trapdoor functions that take an additional random input which cannot always be recovered using the trapdoor (these are, in some sense, "randomized" trapdoor functions). Their constructions result in new CCA-secure public-key encryption schemes that are based on lattices. A similar approach was taken by Dowsley et al. [14] who relied on our framework to constructed the first CCA-secure variant of the coding-based McEliece encryption scheme [38]. This was later improved by Freeman et al. [16] who constructed a collection of injective trapdoor functions that are secure under correlated products based on the hardness of syndrome decoding (this, in particular, implies a CCA-secure public-key encryption scheme). These demonstrate that the correlated products approach for chosen-ciphertext security is fruitful, and that security under correlated products is achievable under a variety of number-theoretic assumptions.

Our framework was also used by Mol and Yilek [39] who demonstrated that even a non-negligible fraction of a single bit of lossiness is sufficient for obtaining chosen-ciphertext secure encryption from lossy trapdoor functions. In terms of the required amount of lossiness this is a significant improvement both to our result and to the result of Peikert and Waters [46]. Specifically, Mol and Yilek show that even "slightly" lossy trapdoor functions are secure under a correlated product which suffices for instantiating our scheme.

The possibility of realizing security under correlated products based on general assumptions was recently studied by Vahlis [52] and by Hemenway, Lu, and Ostrovsky [31]. On the negative side, Vahlis showed that trapdoor permutations do not imply in a black-box manner trapdoor permutation (or injective trapdoor functions) that are secure under correlated products. On the positive side, Hemenway et al. showed that if one is not interested in having a trapdoor, then one-way functions that are secure under correlated product can in fact be constructed from any one-way functions (in a black-box manner). This strengthens the intuition that security under correlated products is significantly weaker than lossiness: our black-box impossibility result in Section 6 rules out in particular black-box constructions of lossy functions (i.e., the same as lossy trapdoor functions but without a trapdoor) from one-way functions.

1.6 Paper Organization

The remainder of the paper is organized as follows. In Section 2 we briefly review several fundamental definitions. In Section 3 we provide a formal treatment of security under correlated products, which is shown to be satisfied by lossy trapdoor functions. In Section 4 we describe a simplified version of our encryption scheme which already illustrates the main ideas underlying our approach. The more general construction is described in Section 5. Finally, in Section 6 we prove that there is no fully-black-box construction of lossy trapdoor functions from trapdoor functions secure under correlated products.

2 Preliminaries

We denote by N the set of all integers, and for an integer $n \in N$ we denote by $[n]$ the set $\{1, \ldots, n\}$. For a finite set X, we denote by $x \leftarrow X$ the experiment of choosing an element of X according to the uniform distribution over X. Similarly, for a distribution $\mathcal{D}$ over a set X, we denote by $x \leftarrow \mathcal{D}$ the experiment of choosing an element of X according to the distribution $\mathcal{D}$.

In the remainder of this section we briefly review the notions of one-way functions, hard-core predicates, trapdoor functions, lossy trapdoor functions, public-key encryption, and one-time signature schemes. We refer the reader to [22, 23, 46] for more elaborated expositions of these notions.

2.1 One-Way Functions and Hard-Core Predicates

Informally, a collection $\mathcal{F}$ of functions is said to be one-way if: (1) it is easy to sample a function f from the collection, (2) given an input x it is easy to compute $f(x)$, and (3) it is computationally infeasible to find a pre-image of $f(x)$ with non-negligible advantage over the choice of x. Typically, it is assumed that x is chosen uniformly at random from the set of all possible inputs, and thus the specification of the exact distribution under which the collection of functions is hard to invert is omitted. However, for the purposes of this paper, it is necessary for us to explicitly specify the input distribution.

Definition 2.1 (Efficiently computable functions)**.** *A collection of* efficiently computable *functions is a pair of probabilistic polynomial-time algorithms $\mathcal{F} = (G, F)$ such that:*

1. *The algorithm G on input 1^n outputs a description $s \in \{0,1\}^n$ of a function $f_s : \{0,1\}^n \to \{0,1\}^n$.*[3]
2. *The algorithm F on input $(s, x) \in \{0,1\}^n \times \{0,1\}^n$ outputs $f_s(x)$.*

Notation 2.2. *Given a collection of function $\mathcal{F} = (G, F)$ and a pair $(s, y) \in \{0,1\}^n \times \{0,1\}^n$, we let $F^{-1}(s, y) = \{x \in \{0,1\}^n \mid y = F(s, x)\}$.*

Definition 2.3 (One-way functions)**.** *Let $\mathcal{I}$ be a distribution where $\mathcal{I}(1^n)$ is distributed over $\{0,1\}^n$. A collection of efficiently computable functions $\mathcal{F} = (G, F)$ is said to be* one-way with respect to the input distribution $\mathcal{I}$ *if for every probabilistic polynomial-time algorithm $\mathcal{A}$ and polynomial $p(\cdot)$, it holds that*

$$\Pr\left[\mathcal{A}(1^n, s, F(s, x)) \in F^{-1}(s, F(s, x))\right] < \frac{1}{p(n)} \quad ,$$

for all sufficiently large n, where $s \leftarrow G(1^n)$ and $x \leftarrow \mathcal{I}(1^n)$.

Definition 2.4 (Hard-core predicate)**.** *Let $\mathcal{I}$ be a distribution where $\mathcal{I}(1^n)$ is distributed over $\{0,1\}^n$, and let $\mathcal{F} = (G, F)$ be a collection of efficiently computable functions. A polynomial-time algorithm $H : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ is said to be a* hard-core predicate of $\mathcal{F}$ with respect to the input distribution $\mathcal{I}$ *if for every probabilistic-polynomial time algorithm $\mathcal{A}$ and polynomial $p(\cdot)$, it holds that*

$$\Pr\left[\mathcal{A}(1^n, s, F(s, x)) = H(s, x)\right] < \frac{1}{2} + \frac{1}{p(n)}$$

for all sufficiently large n, where $s \leftarrow G(1^n)$ and $x \leftarrow \mathcal{I}(1^n)$.

[3] Generally speaking, the input, the output and the description of a function may be of different lengths (though polynomially related). For simplicity, we assume that all three are n-bit strings.

In this paper we focus on injective functions, and in this case the hardness of predicting the value of a predicate from the value of the function implies in particular the hardness of inverting the function. The Goldreich-Levin theorem [25] can be used in our setting (where considering arbitrary input distributions) to guarantee the existence of a hard-core predicate for any collection of one-way functions. The hard-core predicate exists with respect to the same input distribution for which the collection of functions is one-way.

Corollary 2.5. *Let $\mathcal{I}$ be a distribution where $\mathcal{I}(1^n)$ is distributed over $\{0,1\}^n$, and let $\mathcal{F} = (G, F)$ be a collection of efficiently computable injective functions. Then, $\mathcal{F}$ is one-way with respect to $\mathcal{I}$ if and only if $\mathcal{F}$ has a hard-core predicate with respect to $\mathcal{I}$.*

2.2 Injective Trapdoor Functions and Lossy Trapdoor Functions

In the following we define the notions of injective trapdoor functions and lossy trapdoor functions.

Definition 2.6 (Trapdoor functions)**.** *A collection of* injective trapdoor functions *is a triplet of probabilistic polynomial-time algorithms $\mathcal{F} = (G, F, F^{-1})$ such that:*

- *The algorithm G on input 1^n outputs a pair $(s, td) \in \{0,1\}^n \times \{0,1\}^n$.*
- *The pair (G_L, F) is a collection of injective one-way functions, where G_L denotes the left part of the output of G.*
- *For every (s, td) in the range of G and $x \in \{0,1\}^n$, the algorithm F^{-1} on input $(td, F(s, x))$ outputs x.*

Definition 2.7 (Lossy trapdoor functions)**.** *A collection of (n, ℓ)-lossy trapdoor functions is a triplet of probabilistic polynomial-time algorithms (G, F, F^{-1}) such that:*

1. *$G(1^n, \mathsf{injective})$ outputs a pair $(s, td) \in \{0,1\}^n \times \{0,1\}^n$. The algorithm $F(s, \cdot)$ computes an injective function $f_s(\cdot)$ over $\{0,1\}^n$, and $F^{-1}(td, \cdot)$ computes $f_s^{-1}(\cdot)$.*
2. *$G(1^n, \mathsf{lossy})$ outputs $s \in \{0,1\}^n$. The algorithm $F(s, \cdot)$ computes a function $f_s(\cdot)$ over $\{0,1\}^n$ whose image has size at most $2^{n-\ell}$.*
3. *The descriptions of functions (i.e., s-values) sampled using $G(1^n, \mathsf{injective})$ and $G(1^n, \mathsf{lossy})$ are computationally indistinguishable.*

2.3 Public-Key Encryption Schemes

The following definition describes the functionality of a public-key encryption scheme:

Definition 2.8 (Public-key encryption)**.** *A public-key encryption scheme is a triplet (KG, E, D) of probabilistic polynomial-time algorithms such that:*

1. *The key generation algorithm KG receives as input a security parameter 1^n and outputs a public key PK and a secret key SK.*
2. *The encryption algorithm E receives as input a public key PK and a message m (in some implicit message space), and outputs a ciphertext c.*
3. *The decryption algorithm D receives as input a ciphertext c and a secret key SK, and outputs a message m or the symbol $\perp$.*

4. *For any message m it holds that $D(SK, E(PK, m)) = m$ with overwhelming probability over the internal coin tosses of KG, E and D.*

In this paper we consider public-key encryption schemes that are secure against adaptive chosen-ciphertext attacks, defined as follows.

Definition 2.9 (Chosen-ciphertext security)**.** *A public-key encryption scheme (KG, E, D) is said to be CCA2-secure if the advantage of any probabilistic polynomial-time adversary $\mathcal{A}$ in the following interaction is negligible in the security parameter:*

1. *$KG(1^n)$ outputs (PK, SK), and $\mathcal{A}$ is given PK.*
2. *$\mathcal{A}$ may adaptively query a decryption oracle $D(SK, \cdot)$.*
3. *At some point $\mathcal{A}$ outputs two messages m_0 and m_1 with $|m_0| = |m_1|$, and receives a challenge ciphertext $c = E(PK, m_b)$ for a uniformly chosen bit $b \in \{0, 1\}$.*
4. *$\mathcal{A}$ may continue to adaptively query the decryption oracle $D(SK, \cdot)$ on any ciphertext other than the challenge ciphertext.*
5. *Finally, $\mathcal{A}$ outputs a bit b'.*

We say that $\mathcal{A}$ succeeds if $b' = b$, and denote the probability of this event by $\Pr[\mathsf{Success}]$*. The advantage of $\mathcal{A}$ is defined as* $|\Pr[\mathsf{Success}] - 1/2|$.

2.4 Signature Schemes

The following definitions describe the functionality of a signature scheme, and the security notion of *one-time strong unforgeability* that is used in this paper.

Definition 2.10 (Signature scheme)**.** *A signature scheme is a triplet* $(\mathsf{KG}_{\mathsf{sig}}, \mathsf{Sign}, \mathsf{Ver})$ *of probabilistic polynomial-time algorithms such that:*

1. *The key generation algorithm* $\mathsf{KG}_{\mathsf{sig}}$ *receives as input a security parameter* 1^n *and outputs a verification key vk and a signing key sk.*
2. *The signing algorithm* Sign *receives as input a signing key sk and a message m (in some implicit message space), and outputs a signature σ.*
3. *The verification algorithm* Ver *receives as input a verification key vk, a message m, and a signature σ, and outputs a bit $b \in \{0, 1\}$.*
4. *For any message m it holds that* $\mathsf{Ver}(vk, m, \mathsf{Sign}(sk, m)) = 1$ *with overwhelming probability over the internal coin tosses of* $\mathsf{KG}_{\mathsf{sig}}$, Sign *and* Ver.

Definition 2.11 (One-time strong unforgeability)**.** *A signature scheme* $(\mathsf{KG}_{\mathsf{sig}}, \mathsf{Sign}, \mathsf{Ver})$ *is said to be one-time strongly unforgeable if the success probability of any probabilistic polynomial-time adversary $\mathcal{A}$ in the following interaction is negligible in the security parameter:*

1. $\mathsf{KG}_{\mathsf{sig}}(1^n)$ *outputs (vk, sk), and $\mathcal{A}$ is given vk.*
2. *$\mathcal{A}$ may output a message m, and is then given in return* $\sigma = \mathsf{Sign}(sk, m)$*. If $\mathcal{A}$ chooses not to output any message, we set $(m, \sigma) = (\perp, \perp)$.*
3. *$\mathcal{A}$ outputs a pair (m^*, σ^*).*

We say that $\mathcal{A}$ succeeds if $\mathsf{Ver}(vk, m^*, \sigma^*) = 1$ *and $(m^*, \sigma^*) \neq (m, \sigma)$.*

3 Security Under Correlated Products

In this section we formally define the notion of security under correlated products, and demonstrate that the notion is satisfied by any collection of lossy trapdoor functions (with appropriately chosen parameters) for a very natural and useful correlation. We then discuss the exact parameters that are required for our encryption scheme, and the number-theoretic assumptions that are currently known to guarantee such parameters.

A collection of functions is represented as a pair of algorithms $\mathcal{F} = (G, F)$, where G is a generation algorithm used for sampling a description of a function, and F is an evaluation algorithm used for evaluating a function on a given input. The following definition formalizes the notion of a k-wise product which introduces a collection $\mathcal{F}_k$ consisting of all k-tuples of functions from $\mathcal{F}$.

Definition 3.1 (k-wise product)**.** *Let $\mathcal{F} = (G, F)$ be a collection of efficiently computable functions. For any integer k, we define the* k-wise product $\mathcal{F}_k = (G_k, F_k)$ *as follows:*

- *The generation algorithm G_k on input 1^n invokes $G(1^n)$ for k times independently and outputs $(s_1, \dots, s_k)$. That is, a function is sampled from $\mathcal{F}_k$ by independently sampling k functions from $\mathcal{F}$.*
- *The evaluation algorithm F_k on input $(s_1, \dots, s_k, x_1, \dots, x_k)$ invokes F to evaluate each function s_i on x_i. That is, $F_k(s_1, \dots, s_k, x_1, \dots, x_k) = (F(s_1, x_1), \dots, F(s_k, x_k))$.*

The notion of a one-way function asks for a function that is efficiently computable but is hard to invert given the image of a uniformly chosen input. More generally, one can naturally extend this notion to consider one-wayness under any specified input distribution, not necessarily the uniform distribution. That is, informally, we say that a function is one-way with respect to an input distribution $\mathcal{I}$ if it is efficiently computable but hard to invert given the image of a random input sampled according to $\mathcal{I}$ (see Section 2 for a formal definition).

In the context of k-wise products, a rather straightforward argument shows that for any collection $\mathcal{F}$ which is one-way with respect to some input distribution $\mathcal{I}$, the k-wise product $\mathcal{F}_k$ is one-way with respect to the input distribution which samples k independent inputs from $\mathcal{I}$. The following definition formalizes the notion of security under correlated products, where the inputs for $\mathcal{F}_k$ may be correlated.

Definition 3.2 (Security under correlated products)**.** *Let $\mathcal{F} = (G, F)$ be a collection of efficiently computable functions, and let $\mathcal{C}_k$ be a distribution where $\mathcal{C}_k(1^n)$ is distributed over $\{0,1\}^{k \cdot n}$ for some integer $k = k(n)$. We say that $\mathcal{F}$ is* secure under a $\mathcal{C}_k$-correlated product *if $\mathcal{F}_k$ is one-way with respect to the input distribution $\mathcal{C}_k$.*

Correlated products security based on lossy trapdoor functions. We conclude this section by demonstrating that, for an appropriate choice of parameters, any collection of lossy trapdoor functions yields a collection of injective trapdoor functions that is secure under a $\mathcal{C}_k$-correlated product. The input distribution under consideration, $\mathcal{C}_k$, samples a uniformly random input x and outputs k copies of x. We refer to this distribution as the *uniform k-repetition distribution*, and this distribution is the one required for the simplified variant of our encryption scheme, presented in Section 4.

Specifically, given a collection of lossy trapdoor functions $\mathcal{F} = (G, F, F^{-1})$ we define a collection $\mathcal{F}_{\mathsf{inj}}$ of injective trapdoor functions by restricting $\mathcal{F}$ to its injective functions. That is, $\mathcal{F}_{\mathsf{inj}} = (G_{\mathsf{inj}}, F, F^{-1})$ where $G_{\mathsf{inj}}(1^n) = G(1^n, \mathsf{injective})$. We prove the following theorem:

Theorem 3.3. *Let $\mathcal{F} = (G, F, F^{-1})$ be a collection of (n, ℓ)-lossy trapdoor functions. Then, for any integer $k < \frac{n-\omega(\log n)}{n-\ell}$, for any probabilistic polynomial-time algorithm $\mathcal{A}$ and polynomial $p(\cdot)$, it holds that*

$$\Pr\left[\mathcal{A}(1^n, s_1, \ldots, s_k, F(s_1, x), \ldots, F(s_k, x)) = x\right] < \frac{1}{p(n)} \ ,$$

for all sufficiently large n, where the probability is taken over the choices of $s_1 \leftarrow G_{\mathsf{inj}}(1^n), \ldots, s_k \leftarrow G_{\mathsf{inj}}(1^n)$, $x \leftarrow \{0,1\}^n$, and over the internal coin tosses of $\mathcal{A}$.

Proof. Peikert and Waters [46, Lemma 3.1] proved that any collection of $(n, \omega(\log n))$-lossy trapdoor functions is one-way. Thus, it is sufficient to prove that $\mathcal{F}_k$ is a collection of $(n, \omega(\log n))$-lossy trapdoor functions. For any k functions $s_1, \ldots, s_k$ sampled according to $G_{\mathsf{inj}}(1^n)$, the function $F_k(s_1, \ldots, s_k, x_1, \ldots, x_k) = (F(s_1, x_1), \ldots, F(s_k, x_k))$ is injective. For any k functions $s_1, \ldots, s_k$ sampled according to $G_{\mathsf{lossy}}(1^n)$, the function $F_k(s_1, \ldots, s_k, x_1, \ldots, x_k) = (F(s_1, x_1), \ldots, F(s_k, x_k))$ obtains at most $2^{k(n-\ell)}$ values, which is upper bounded by $2^{n-\omega(\log n)}$ for any $k < \frac{n-\omega(\log n)}{n-\ell}$. Finally, note that a standard hybrid argument shows that the distribution obtained by independently sampling k functions according to $G_{\mathsf{inj}}(1^n)$ is computationally indistinguishable from the distribution obtained by independently sampling k functions according to $G_{\mathsf{lossy}}(1^n)$. Thus, $\mathcal{F}_k$ is a collection of $(n, \omega(\log n))$-lossy trapdoor functions. ■

The required parameters for our scheme. The assumption underlying our encryption scheme asks for $k(n) = \omega(\log n)$ for CCA1 security, and for $k(n) = n^{\epsilon}$ (for some constant $0 < \epsilon < 1$) for CCA2 security. In turn, existing constructions of lossy trapdoor functions guaranteing these parameters [1, 46, 49] imply that our encryption scheme can be realized under the hardness of the decisional Diffie-Hellman problem, and of Paillier's decisional composite residuosity problem. We note that the lattice-based construction of Peikert and Waters [46] guarantees only a constant $k(n)$ that is not sufficient for our encryption scheme. However, Peikert [45] and Goldwasser and Vaikuntanathan [27] recently showed that security under correlated products (with sufficiently large $k(n)$) is nevertheless achievable under the worst-case hardness of lattice problems, although these are currently known to imply lossy trapdoor functions with only a relatively small amount of loss.

4 A Simplified Construction

In this section we describe a simplified version of our construction which already illustrates the main ideas underlying our approach. The encryption scheme presented in the current section is a simplification in the sense that it relies on a seemingly stronger computational assumption than the more generalized construction which is presented in Section 5. In addition, we first present the scheme as encrypting only one bit messages, and then demonstrate that it naturally extends to multi-bit messages. In what follows we state the computational assumption, describe the encryption scheme, prove its security, and describe the extension to multi-bit messages.

The underlying computational assumption. The computational assumption underlying the simplified scheme is that there exists a collection $\mathcal{F}$ of injective trapdoor functions and an integer function $k = k(n)$ such that $\mathcal{F}$ is secure under a $\mathcal{C}_k$-correlated product, where $\mathcal{C}_k$ is the uniform k-repetition distribution (i.e., outputs k copies of a uniformly distributed input x). Specifically, our scheme uses a hard-core predicate $h : \{0,1\}^* \to \{0,1\}$ for $\mathcal{F}_k$ with respect to $\mathcal{C}_k$. That is, the underlying computational assumption is that for any probabilistic polynomial-time predictor $\mathcal{P}$ it holds that

$$\left|\Pr\left[\mathcal{P}(1^n, s_1, \ldots, s_k, F(s_1, x), \ldots, F(s_k, x)) = h(s_1, \ldots, s_k, x)\right] - \frac{1}{2}\right|$$

is negligible in n, where the probability is taken over the choices of $s_1 \leftarrow G(1^n), \ldots, s_k \leftarrow G(1^n)$, $x \leftarrow \{0,1\}^n$, and over the internal coin tosses of $\mathcal{P}$.

The integer function $k(n)$ should correspond to the bit-length of verification keys of some one-time strongly-unforgeable signature scheme $(\mathsf{KG_{sig}}, \mathsf{Sign}, \mathsf{Ver})$. By applying a universal one-way hash function to the verification keys (as in [13]) it suffices that the above assumption holds for $k(n) = n^\epsilon$ for a constant $0 < \epsilon < 1$. For simplicity, however, when describing our scheme we do not apply a universal one-way hash function to the verification keys. We also note that for an even more simplified version which is only CCA1-secure (the one described in Section 1.3), any $k(n) = \omega(\log n)$ suffices.

The construction. The following describes our simplified encryption scheme given by the triplet (KG, E, D).

- **Key generation:** On input 1^n the algorithm invokes $G(1^n)$ for $2k$ times independently to obtain $2k$ descriptions of functions from $\mathcal{F}$ denoted $(s_1^0, s_1^1), \ldots, (s_k^0, s_k^1)$ together with the corresponding trapdoors $(td_1^0, td_1^1), \ldots, (td_k^0, td_k^1)$. The public-key and secret-key are defined as follows:

$$\begin{aligned} PK &= \left(\left(s_1^0, s_1^1\right), \ldots, \left(s_k^0, s_k^1\right)\right) \\ SK &= \left(\left(td_1^0, td_1^1\right), \ldots, \left(td_k^0, td_k^1\right)\right) \ . \end{aligned}$$

- **Encryption:** On input a message $m \in \{0,1\}$ and a public key PK, the algorithm samples $(vk, sk) \leftarrow \mathsf{KG_{sig}}(1^n)$ where $vk = vk_1 \circ \cdots \circ vk_k \in \{0,1\}^k$, chooses a uniformly distributed $x \in \{0,1\}^n$, and outputs $(vk, y_1, \ldots, y_k, c_1, c_2)$ where

$$\begin{aligned} y_i &= F\left(s_i^{vk_i}, x\right) \quad \forall i \in [k] \\ c_1 &= m \oplus h\left(s_1^{vk_1}, \ldots, s_k^{vk_k}, x\right) \\ c_2 &= \mathsf{Sign}(sk, (y_1, \ldots, y_k, c_1)) \ . \end{aligned}$$

- **Decryption:** On input a ciphertext $(vk, y_1, \ldots, y_k, c_1, c_2)$ and a secret-key SK, the algorithm acts as follows. If $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 0$, it outputs $\perp$. Otherwise, for every $i \in [k]$ it computes $x_i = F^{-1}\left(td_i^{vk_i}, y_i\right)$. If $x_1 = \cdots = x_k$ then it outputs $c_1 \oplus h\left(s_1^{vk_1}, \ldots, s_k^{vk_k}, x_1\right)$, and otherwise it outputs $\perp$.

The following theorem establishes the security of the scheme.

Theorem 4.1. *Assuming that $\mathcal{F}$ is secure under a $\mathcal{C}_k$-correlated product, where $\mathcal{C}_k$ is the uniform k-repetition distribution, and that $(\mathsf{KG_{sig}}, \mathsf{Sign}, \mathsf{Ver})$ is one-time strongly unforgeable, the encryption scheme (KG, E, D) is CCA2-secure.*

Proof. Let $\mathcal{A}$ be a probabilistic polynomial-time CCA2-adversary (see Definition 2.9). We denote by Forge the event in which for one of $\mathcal{A}$'s decryption queries $(vk, y_1, \ldots, y_k, c_1, c_2)$ during the CCA2 interaction it holds that $vk = vk^*$ where vk^* is the verification key used in the challenge ciphertext, and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$. We first argue that the event Forge has a negligible probability due to the security of the one-time signature scheme. Then, we construct a probabilistic polynomial-time algorithm $\mathcal{P}$ that predicts the hard-core predicate h while essentially preserving the advantage of $\mathcal{A}$.

More formally, we denote by $\mathsf{Success}$ the event in which $\mathcal{A}$ successfully guesses the bit b used for encrypting the challenge ciphertext. Then, the advantage of $\mathcal{A}$ in the CCA2 interaction is bounded as follows:

$$\begin{aligned}\left|\Pr\left[\mathsf{Success}\right]-\frac{1}{2}\right| &\leq \left|\Pr\left[\mathsf{Success}\wedge\mathsf{Forge}\right]-\frac{1}{2}\Pr\left[\mathsf{Forge}\right]\right|+\left|\Pr\left[\mathsf{Success}\wedge\overline{\mathsf{Forge}}\right]+\frac{1}{2}\Pr\left[\mathsf{Forge}\right]-\frac{1}{2}\right| \\ &\leq \frac{1}{2}\Pr\left[\mathsf{Forge}\right]+\left|\Pr\left[\mathsf{Success}\wedge\overline{\mathsf{Forge}}\right]+\frac{1}{2}\Pr\left[\mathsf{Forge}\right]-\frac{1}{2}\right| \enspace .\end{aligned}$$

The theorem follows from the following two claims:

Claim 4.2. $\Pr\left[\mathsf{Forge}\right]$ *is negligible.*

Proof. We show that any probabilistic polynomial-time adversary $\mathcal{A}$ for which $\Pr\left[\mathsf{Forge}\right]$ is non-negligible, can be used to construct a probabilistic polynomial-time adversary $\mathcal{A}'$ that breaks the security of the one-time signature scheme with the same probability. The adversary $\mathcal{A}'$ is given a verification key vk^* sampled using $\mathsf{KG_{sig}}(1^n)$ and simulates the CCA2 interaction to $\mathcal{A}$ as follows. $\mathcal{A}'$ begins by invoking the key generation algorithm on input 1^n and gives the public-key to $\mathcal{A}$. Whenever $\mathcal{A}$ submits a decryption query $(vk, y_1, \ldots, y_k, c_1, c_2)$, $\mathcal{A}'$ acts as follows. If $vk = vk^*$ and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$, then $\mathcal{A}'$ outputs $((y_1, \ldots, y_k, c_1), c_2)$ as the forgery and halts. Otherwise, $\mathcal{A}'$ invokes the decryption procedure. In the challenge phase, upon receiving two message m_0 and m_1, $\mathcal{A}'$ chooses $b \in \{0,1\}$ and $x \in \{0,1\}^n$ uniformly at random, and computes

$$\begin{aligned} y_i &= F\left(s_i^{vk_i^*}, x\right) \quad \forall i \in [k] \\ c_1 &= m_b \oplus h\left(s_1^{vk_1^*}, \ldots, s_k^{vk_k^*}, x\right) \enspace . \end{aligned}$$

Then, it obtains a signature c_2 on $(y_1, \ldots, y_k, c_1)$ with respect to vk^* (recall that $\mathcal{A}'$ is allowed to ask for a signature on one message). Finally, it sends $(vk^*, y_1, \ldots, y_k, c_1, c_2)$ to $\mathcal{A}$. We note that during the second decryption phase, if $\mathcal{A}$ submits the challenge ciphertext as a decryption query, then $\mathcal{A}'$ responds with $\perp$.

Note that prior to the first decryption query in which Forge occurs (assuming that Forge indeed occurs), the simulation of the CCA2 interaction is perfect. Therefore, the probability that $\mathcal{A}'$ breaks the security of the one-time signature scheme is exactly $\Pr\left[\mathsf{Forge}\right]$. The security of the signature scheme implies that this probability is negligible. ∎

Claim 4.3. $\left|\Pr\left[\mathsf{Success}\wedge\overline{\mathsf{Forge}}\right]+\frac{1}{2}\Pr\left[\mathsf{Forge}\right]-\frac{1}{2}\right|$ *is negligible.*

Proof. Given a probabilistic polynomial-time adversary $\mathcal{A}$, we construct a probabilistic polynomial-time predictor $\mathcal{P}$ for the hard-core predicate h. We show that the advantage of $\mathcal{P}$ is exactly $\left|\Pr\left[\mathsf{Success}\wedge\overline{\mathsf{Forge}}\right]+\frac{1}{2}\Pr\left[\mathsf{Forge}\right]-\frac{1}{2}\right|$, and this is assumed to be negligible by the unpredictability of h. Recall that the advantage of $\mathcal{P}$ is defined as

$$\left|\Pr\left[\mathcal{P}(1^n, s_1, \ldots, s_k, F(s_1, x), \ldots, F(s_k, x)) = h(s_1, \ldots, s_k, x)\right]-\frac{1}{2}\right| \enspace ,$$

where $s_1 \leftarrow G(1^n), \ldots, s_k \leftarrow G(1^n)$ independently, and the probability is taken over the uniform choice of $x \in \{0,1\}^n$, and over the internal coin tosses of both G and $\mathcal{P}$.

For simplicity, we first construct an efficient distinguisher $\mathcal{A}'$ which receives input of the form $(1^n, s_1, \ldots, s_k, F(s_1, x), \ldots, F(s_k, x))$ and a bit $w \in \{0,1\}$ which is either $h(s_1, \ldots, s_k, x)$ or a uniformly random bit, and is able to distinguish between the two cases with a non-negligible probability. The distinguisher $\mathcal{A}'$ acts by simulating the CCA2 interaction to $\mathcal{A}$. More specifically, on input $(1^n, s_1, \ldots, s_k, y_1, \ldots, y_k)$ and a bit w, the distinguisher $\mathcal{A}'$ first creates a pair (PK, SK) as follows. It samples $(vk^*, sk^*) \leftarrow \mathsf{KG}_{\mathsf{sig}}(1^n)$, where $vk^* = vk_1^* \circ \cdots \circ vk_k^* \in \{0,1\}^k$, and for every $i \in [k]$ sets $s_i^{vk_i^*} = s_i$ and samples $\left(s_i^{1-vk_i^*}, td_i^{1-vk_i^*}\right) \leftarrow G(1^n)$. Then, $\mathcal{A}'$ outputs the public-key

$$PK = \left(\left(s_1^0, s_1^1\right), \ldots, \left(s_k^0, s_k^1\right)\right) \ .$$

In the decryption phases, whenever $\mathcal{A}$ submits a decryption query of the form $(vk, y_1, \ldots, y_k, c_1, c_2)$, $\mathcal{A}'$ acts as follows:

1. If $vk = vk^*$ and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$ (note that this means that the event Forge occurs), then it outputs an independently and uniformly chosen bit and halts.

2. If $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 0$, then it responds with $\perp$.

3. If $vk \neq vk^*$ and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$, then it picks some $i \in [k]$ for which $vk_i \neq vk_i^*$ and computes $x = F^{-1}\left(td_i^{vk_i}, y_i\right)$. If for every $j \in [k]$ it holds that $y_j = F\left(s_j^{vk_j}, x\right)$, it responds with $c_1 \oplus h\left(s_1^{vk_1}, \ldots, s_k^{vk_k}, x\right)$, and otherwise it responds with $\perp$.

In the challenge phase, given two messages m_0 and m_1, $\mathcal{A}'$ chooses a random bit $b \in \{0,1\}$ and replies with the challenge ciphertext

$$c = (vk^*, y_1, \ldots, y_k, c_1, c_2) \ ,$$

where $c_1 = m_b \oplus w$, and $c_2 = \mathsf{Sign}(sk^*, (y_1, \ldots, y_k, c_1))$. We note that during the second decryption phase, if $\mathcal{A}$ submits the challenge ciphertext as a decryption query, then $\mathcal{A}'$ responds with $\perp$. At the end of this interaction $\mathcal{A}$ outputs a bit b' (assuming that $\mathcal{A}'$ did not halt due to the event Forge). If $b' = b$ then $\mathcal{A}'$ outputs 1, and otherwise $\mathcal{A}'$ outputs 0.

In order to compute the advantage of $\mathcal{A}'$ in distinguishing between $h(s_1, \ldots, s_k, x)$ and a uniformly random bit, we observe the following:

1. If w is a uniformly random bit, then the challenge ciphertext in the simulated interaction is independent of b. Therefore, independently of whether the event Forge occurs or does not occur, the probability that $\mathcal{A}'$ outputs 1 in this case is exactly $1/2$.

2. If $w = h(s_1, \ldots, s_k, x)$ we consider two cases. If the event Forge occurs, then clearly $\mathcal{A}'$ outputs 1 with probability $1/2$. In addition, if the event Forge does not occur, the simulated interaction is identical to the CCA2 interaction (a formal argument follows). Therefore, the probability that $\mathcal{A}'$ outputs 1 in this case is exactly $\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr\left[\mathsf{Forge}\right]$.

   Note that the only difference between the CCA2 interaction and the simulated interaction is the distribution of the challenge ciphertext: In the CCA2 interaction the value vk in the challenge ciphertext is a randomly chosen verification key, and in the simulated interaction the value vk is chosen ahead of time by $\mathcal{A}$. In what follows we claim that as long as the event Forge does not occur, the distribution of vk in the challenge ciphertext is identical in the two cases.

Formally, denote by $vk_1, \ldots, vk_q$ the random variables corresponding to the value of vk in $\mathcal{A}$'s decryption queries (without loss of generality we assume that $\mathcal{A}$ always submits q queries, and that the signature verification never fails on these queries). In the CCA2 interaction, as long as the event Forge does not occur, it holds that the verification key used for the challenge ciphertext is a random verification key with the only restriction that it is different than $vk_1, \ldots, vk_q$. In the simulated interaction, given that $vk^* \notin \{vk_1, \ldots, vk_q\}$, we claim that from $\mathcal{A}$'s point of view, the value vk^* is also a random verification key which is different than $vk_1, \ldots, vk_q$. That is, given that $vk^* \notin \{vk_1, \ldots, vk_q\}$, the distribution of the resulting transcript is independent of the specific value of vk^*.

Indeed, first note that the public key is independent of vk^*. Now consider a decryption query $(vk, y_1, \ldots, y_k, c_1, c_2)$ for some $vk \in \{vk_1, \ldots, vk_q\}$. For any $vk^* \neq vk$, if $y_1, \ldots, y_k$ have the same preimage x, then the decryption algorithm will always output $c_1 \oplus h\left(s_1^{vk_1}, \ldots, s_k^{vk_k}, x\right)$. In addition, for any $vk^* \neq vk$, if $y_1, \ldots, y_k$ do not have the same preimage, then the decryption algorithm will always output $\perp$.

The above observations imply that

$$\begin{aligned}&\left|\Pr\left[\mathcal{A}' \text{ outputs } 1 \mid w = h(s_1, \ldots, s_k, x)\right] - \Pr\left[\mathcal{A}' \text{ outputs } 1 \mid w \text{ is random}\right]\right| \\ &\quad = \left|\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr\left[\mathsf{Forge}\right] - \frac{1}{2}\right| \ .\end{aligned}$$

A standard argument (see, for example, [22, Chapter 3.4]) can be applied to efficiently transform $\mathcal{A}'$ into a predictor $\mathcal{P}$ that predicts $h(s_1, \ldots, s_k, x)$ with the same probability. ■

This completes the proof of Theorem 4.1. ■

Encrypting any polynomial number of bits. For simplicity we presented the encryption scheme above for one-bit plaintexts. As recently shown by Myers and shelat [40] it is possible to generically transform any single-bit CCA-secure encryption scheme into a multi-bit one.[4] We now explain how our approach extends to plaintexts of any polynomial length directly, while relying on the same computational assumption.

Recall that the underlying computational assumption is the existence of a collection $\mathcal{F}$ of injective trapdoor functions and an integer function $k = k(n)$ such that $\mathcal{F}_k$ is one-way under the uniform k-repetition distribution (i.e., $x_1 = \cdots = x_k$ where x_1 is chosen uniformly at random). Specifically, the scheme uses a hard-core predicate $h : \{0,1\}^* \to \{0,1\}$ for $\mathcal{F}_k$ to mask the plaintext bit. This assumption clearly implies that for any polynomial $T = T(n)$ there exists a collection $\mathcal{F}'$ of injective trapdoor functions such that $\mathcal{F}'$ is one-way under the uniform k-repetition distribution, and has a hard-core *function* $h' : \{0,1\}^* \to \{0,1\}^T$ that can be used in our scheme to mask T-bit plaintexts. Specifically, the collection $\mathcal{F}'$ is defined as follows: for every function $f : \{0,1\}^n \to \{0,1\}^m$ in $\mathcal{F}$ define a function $f' : \{0,1\}^{Tn} \to \{0,1\}^{Tm}$ by $f'(x_1, \ldots x_T) = (f(x_1), \ldots, f(x_T))$. The security proof of the T-bit encryption scheme is essentially identical to the proof of Theorem 4.1 by showing that any successful CCA-adversary can be used to either break the one-time signature scheme or to break the pseudorandomness of h'.

The above transformation can be viewed as a concatenation of bit encryptions that are dependent in the sense that they share the same verification key vk in the concatenated ciphertext. The

[4] In the case of semantic security under a chosen-plaintext attack it is straightforward to construct a multi-bit encryption scheme from any one-bit encryption scheme by independently encrypting the individual bits of the plaintext. For semantic security under an adaptive chosen-ciphertext attack (CCA2), however, this approach fails.

fact that it retains CCA security is interesting in light of the fact that independently concatenating bit encryptions does not preserve CCA2 security. Our transformation incurs an efficiency loss and consequently security deterioration. Our efficiency loss is smaller than the one incurred by the one of Myers and shelat [40], and specifically, in our case the deterioration is linear in the number of encrypted bits, whereas in their case it is polynomial. An interesting question would be to find an alternative transformation that does not suffer from this shortcoming.

5 The Full-Fledged Construction

In this section we present a more generalized variant of the encryption scheme presented in Section 4. The construction is based on the following ingredients:

1. A collection $\mathcal{F} = (G, F, F^{-1})$ of injective trapdoor functions which is secure under a $\mathcal{C}_k$-correlated product, where $\mathcal{C}_k$ can be any input distribution with the following property: Any $(x_1, \dots, x_k)$ in the support of $\mathcal{C}_k(1^n)$ can be reconstructed given any $t = (1-\epsilon)k$ entries from $(x_1, \dots, x_k)$, for some $0 < \epsilon < 1$. The simplified construction from Section 4 represents the case $t = 1$.

   Specifically, our scheme uses a hard-core predicate $h : \{0,1\}^* \to \{0,1\}$ for the collection $\mathcal{F}$ with respect to $\mathcal{C}_k$. That is, we assume that for any probabilistic polynomial-time predictor $\mathcal{P}$ it holds that

   $$\left|\Pr\left[\mathcal{P}(1^n, s_1, \dots, s_k, F(s_1, x_1), \dots, F(s_k, x_k)) = h(s_1, \dots, s_k, x_1, \dots, x_k)\right] - \frac{1}{2}\right|$$

   is negligible in n, where the probability is taken over the choices of $s_1 \leftarrow G(1^n), \dots, s_k \leftarrow G(1^n)$, $(x_1, \dots, x_k) \leftarrow \mathcal{C}_k(1^n)$, and over the internal coin tosses of $\mathcal{P}$.

2. An error-correcting code $ECC : \Sigma^\ell \to \Sigma^k$ with distance t and polynomial-time encoding (where Σ is an appropriately chosen finite alphabet).

3. A strongly-unforgeable one-time signature scheme $(\mathsf{KG_{sig}}, \mathsf{Sign}, \mathsf{Ver})$. For simplicity we assume that verification keys are elements of Σ^ℓ (we implicitly assume the existence of any injective mapping from the set of verification keys to Σ^ℓ). As mentioned in Section 4, it is possible to apply a universal one-way hash function to the verification keys to improve the efficiency of the scheme.

The following describes the encryption scheme given by the triplet (KG, E, D). For simplicity we consider only one-bit messages, and note that it naturally extends to multi-bit messages, as in Section 4.

- **Key generation:** On input 1^n the algorithm invokes $G(1^n)$ for $k \cdot |\Sigma|$ times independently to obtain $k \cdot |\Sigma|$ descriptions of functions from $\mathcal{F}$ denoted $\{s_1^\sigma\}_{\sigma\in\Sigma}, \dots, \{s_k^\sigma\}_{\sigma\in\Sigma}$ together with the corresponding trapdoors $\{td_1^\sigma\}_{\sigma\in\Sigma}, \dots, \{td_k^\sigma\}_{\sigma\in\Sigma}$. The public key and secret key are defined as follows:

  $$\begin{aligned} PK &= \left(\{s_1^\sigma\}_{\sigma\in\Sigma}, \dots, \{s_k^\sigma\}_{\sigma\in\Sigma}\right) \\ SK &= \left(\{td_1^\sigma\}_{\sigma\in\Sigma}, \dots, \{td_k^\sigma\}_{\sigma\in\Sigma}\right) \ , \end{aligned}$$

- **Encryption:** On input a message $m \in \{0,1\}$ and a public key PK, the algorithm samples $(vk, sk) \leftarrow \mathsf{KG_{sig}}(1^n)$ and $(x_1, \ldots, x_k) \leftarrow \mathcal{C}_k(1^n)$. Then, it computes $ECC(vk) = \sigma_1 \circ \cdots \circ \sigma_k$, and outputs $c = (vk, y_1, \ldots, y_k, c_1, c_2)$ where

$$\begin{aligned} y_i &= F\left(s_i^{\sigma_i}, x_i\right) \quad \forall i \in [k] \\ c_1 &= m \oplus h\left(s_1^{\sigma_1}, \ldots, s_k^{\sigma_k}, x_1, \ldots, x_k\right) \\ c_2 &= \mathsf{Sign}\left(sk, (y_1, \ldots, y_k, c_1)\right) \quad . \end{aligned}$$

- **Decryption:** On input a ciphertext $c = (vk, y_1, \ldots, y_k, c_1, c_2)$ and a secret key SK, the algorithm acts as follows. If $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 0$, it outputs $\perp$. Otherwise, the algorithm picks some distinct $i_1, \ldots, i_t \in [k]$, computes

$$\begin{aligned} x_{i_1} &= F^{-1}\left(td_{i_1}^{\sigma_{i_1}}, y_{i_1}\right) \\ &\vdots \\ x_{i_t} &= F^{-1}\left(td_{i_t}^{\sigma_{i_t}}, y_{i_t}\right) \quad , \end{aligned}$$

and uses the values $(i_1, x_{i_1}), \ldots, (i_t, x_{i_t})$ to reconstruct the unique tuple $(x_1, \ldots, x_k)$ in the support of $\mathcal{C}_k(1^n)$ which is consistent with $(i_1, x_{i_1}), \ldots, (i_t, x_{i_t})$. Finally, if for every $j \in [k]$ it holds that $y_j = F\left(s_j^{\sigma_j}, x_j\right)$, then it outputs $c_1 \oplus h(s_1^{\sigma_1}, \ldots, s_k^{\sigma_k}, x_1, \ldots, x_k)$. Otherwise, it outputs $\perp$.

The following theorem establishes the security of the scheme (KG, E, D).

Theorem 5.1. *Assuming that $\mathcal{F}$ is secure under a $\mathcal{C}_k$-correlated product, and that the signature scheme $(\mathsf{KG_{sig}}, \mathsf{Sign}, \mathsf{Ver})$ is one-time strongly unforgeable, the encryption scheme (KG, E, D) is CCA2-secure.*

Proof. The proof is analogous to that of Theorem 4.1. Given a probabilistic polynomial-time CCA2-adversary $\mathcal{A}$, we denote by Forge the event in which for one of $\mathcal{A}$'s decryption queries $(vk, y_1, \ldots, y_k, c_1, c_2)$ during the CCA2 interaction it holds that $vk = vk^*$ where vk^* is the verification key used in the challenge ciphertext, and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$. Denote by $\mathsf{Success}$ the event in which $\mathcal{A}$ successfully guesses the bit b used for encrypting the challenge ciphertext. As in the proof of Theorem 4.1, the advantage of $\mathcal{A}$ in the CCA2 interaction can be bounded as follows:

$$\begin{aligned} \left|\Pr\left[\mathsf{Success}\right] - \frac{1}{2}\right| &\leq \left|\Pr\left[\mathsf{Success} \wedge \mathsf{Forge}\right] - \frac{1}{2}\Pr\left[\mathsf{Forge}\right]\right| + \left|\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr\left[\mathsf{Forge}\right] - \frac{1}{2}\right| \\ &\leq \frac{1}{2}\Pr\left[\mathsf{Forge}\right] + \left|\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr\left[\mathsf{Forge}\right] - \frac{1}{2}\right| \quad . \end{aligned}$$

Claim 5.2. $\Pr\left[\mathsf{Forge}\right]$ *is negligible.*

Proof. The proof is essentially identical to the proof of Claim 4.2. We show that any probabilistic polynomial-time adversary $\mathcal{A}$ for which $\Pr\left[\mathsf{Forge}\right]$ is non-negligible, can be used to construct a probabilistic polynomial-time adversary $\mathcal{A}'$ that breaks the security of the one-time signature with the same probability. The adversary $\mathcal{A}'$ is given a verification key vk^* sampled using $\mathsf{KG_{sig}}(1^n)$ and simulates the CCA2 interaction to $\mathcal{A}$ as follows. $\mathcal{A}'$ begins by invoking the key generation algorithm on input 1^n and using vk^* for forming the public and secret keys. In the decryption phases, whenever $\mathcal{A}$ submits a decryption query $(vk, y_1, \ldots, y_k, c_1, c_2)$, $\mathcal{A}'$ acts as follows. If $vk = vk^*$ and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$, then $\mathcal{A}'$ outputs $((y_1, \ldots, y_k, c_1), c_2)$ as the forgery and halts. Otherwise, $\mathcal{A}'$ invokes the decryption procedure. In the challenge phase, upon receiving two message m_0 and m_1, $\mathcal{A}'$ computes $ECC(vk^*) = \sigma_1 \circ \cdots \circ \sigma_k$, chooses $b \in \{0,1\}$ and $x \in \{0,1\}^n$ uniformly at random, and computes

$$\begin{aligned} y_i &= F\left(s_i^{\sigma_i}, x_i\right) \ \ \forall i \in [k] \\ c_1 &= m \oplus h\left(s_1^{\sigma_1}, \ldots, s_k^{\sigma_k}, x_1, \ldots, x_k\right) \ . \end{aligned}$$

Then, it obtains a signature c_2 on $(y_1, \ldots, y_k, c_1)$ with respect to vk^* (recall that $\mathcal{A}'$ is allowed to ask for a signature on one message). Finally, it sends $(vk^*, y_1, \ldots, y_k, c_1, c_2)$ to $\mathcal{A}$. We note that during the second decryption phase, if $\mathcal{A}$ submits the challenge ciphertext as a decryption query, then $\mathcal{A}'$ responds with $\perp$.

Note that prior to the first decryption query in which Forge occurs (assuming that Forge indeed occurs), the simulation of the CCA2 interaction is perfect. Therefore, the probability that $\mathcal{A}'$ breaks the security of the one-time signature scheme is exactly $\Pr[\mathsf{Forge}]$. The security of the signature scheme implies that this probability is negligible. ∎

Claim 5.3. $\left|\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right|$ *is negligible.*

Proof. The proof is almost identical to the proof of Claim 4.3, which uses $\mathcal{A}$ to guess the hard-core predicate h. The only technical difference is in arguing that whenever the event Forge does not occur and $w = h(s_1, \ldots, s_k, x_1, \ldots, x_k)$, the simulated interaction is identical to the CCA2 interaction. The argument, however, is still very similar, and is based on the fact that any decryption query in which vk is different than vk^* does not reveal any information on vk^*.

Given a probabilistic polynomial-time adversary $\mathcal{A}$, we construct a probabilistic polynomial-time predictor $\mathcal{P}$ for the hard-core predicate h. We show that the advantage of $\mathcal{P}$ is exactly $\left|\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right|$, and this is assumed to be negligible by the unpredictability of h. Recall that the advantage of $\mathcal{P}$ is defined as

$$\left|\Pr\left[\mathcal{P}(1^n, s_1, \ldots, s_k, F(s_1, x_1), \ldots, F(s_k, x_k)) = h(s_1, \ldots, s_k, x_1, \ldots, x_k)\right] - \frac{1}{2}\right| \ ,$$

where $s_1 \leftarrow G(1^n), \ldots, s_k \leftarrow G(1^n)$, $(x_1, \ldots, x_k) \leftarrow \mathcal{C}_k(1^n)$, and the probability is taken also over the internal coin tosses of $\mathcal{P}$.

Similarly to the proof of Claim 4.2, it will be sufficient to construct an efficient distinguisher $\mathcal{A}'$ which receives input of the form $(1^n, s_1, \ldots, s_k, F(s_1, x_1), \ldots, F(s_k, x_k))$ and a bit $w \in \{0,1\}$ which is either $h(s_1, \ldots, s_k, x_1, \ldots, x_k)$ or a uniformly random bit, and is able to distinguish between the two cases with a non-negligible probability. The distinguisher $\mathcal{A}'$ acts by simulating the CCA2 interaction to $\mathcal{A}$. More specifically, on input $(1^n, s_1, \ldots, s_k, y_1, \ldots, y_k)$ and a bit w, the distinguisher $\mathcal{A}'$ first creates a pair (PK, SK) as follows. It samples $(vk^*, sk^*) \leftarrow \mathsf{KG}_{\mathsf{sig}}(1^n)$ and computes $ECC(vk^*) = \sigma_1^* \circ \cdots \circ \sigma_k^*$. Then, for every $i \in [k]$ it sets $s_i^{\sigma_i^*} = s_i$ and samples $(s_i^\sigma, td_i^\sigma) \leftarrow G(1^n)$ for every $\sigma \in \Sigma$ such that $\sigma \neq \sigma_i^*$. The resulting public-key is

$$PK = \left(\{s_1^\sigma\}_{\sigma \in \Sigma}, \ldots, \{s_k^\sigma\}_{\sigma \in \Sigma}\right) \ .$$

In the decryption phases, whenever $\mathcal{A}$ submits a decryption query $(vk, y_1, \ldots, y_k, c_1, c_2)$, $\mathcal{A}'$ acts as follows:

1. If $vk = vk^*$ and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$ (note that this means that the event Forge occurs), then it outputs an independently and uniformly chosen bit and halts.

2. If $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 0$, then it responds with $\perp$.

3. If $vk \neq vk^*$ and $\mathsf{Ver}(vk, (y_1, \ldots, y_k, c_1), c_2) = 1$, since ECC has distance t, there must exist some distinct $i_1, \ldots, i_t \in [k]$ for which $\sigma_i \neq \sigma_i^*$. This means that $\mathcal{A}'$ is able to compute

$$x_{i_1} = F^{-1}\left(td_{i_1}^{\sigma_{i_1}}, y_{i_1}\right)$$
$$\vdots$$
$$x_{i_t} = F^{-1}\left(td_{i_t}^{\sigma_{i_t}}, y_{i_t}\right) \quad ,$$

and then use the values $(i_1, x_{i_1}), \ldots, (i_t, x_{i_t})$ to reconstruct the unique tuple $(x_1, \ldots, x_k)$ in the support of $\mathcal{C}_k(1^n)$ which is consistent with $(i_1, x_{i_1}), \ldots, (i_t, x_{i_t})$. Finally, if for every $j \in [k]$ it holds that $y_j = F\left(s_j^{\sigma_j}, x_j\right)$, then it responds with $c_1 \oplus h(s_1^{\sigma_1}, \ldots, s_k^{\sigma_k}, x_1, \ldots, x_k)$, and otherwise it responds with $\perp$.

In the challenge phase, given two messages m_0 and m_1, $\mathcal{A}'$ chooses a random bit $b \in \{0, 1\}$ and replies with the challenge ciphertext

$$c = (vk^*, y_1, \ldots, y_k, c_1, c_2) \quad ,$$

where $c_1 = m_b \oplus w$, and $c_2 = \mathsf{Sign}(sk^*, (y_1, \ldots, y_k, c_1))$. Note that during the second decryption phase, if $\mathcal{A}$ submits the challenge ciphertext as a decryption query, then $\mathcal{A}'$ responds with $\perp$. At the end of this interaction $\mathcal{A}$ outputs a bit b'. If $b' = b$ then $\mathcal{A}'$ outputs 1, and otherwise $\mathcal{A}'$ outputs 0.

In order to compute the advantage of $\mathcal{A}'$ in distinguishing between $h(s_1, \ldots, s_k, x_1, \ldots, x_k)$ and a uniformly random bit, we observe the following:

1. If w is a uniformly random bit, then the challenge ciphertext in the simulated interaction is independent of b. Therefore, independently of whether the event Forge occurs or does not occur, the probability that $\mathcal{A}'$ outputs 1 in this case is exactly 1/2.

2. If $w = h(s_1, \ldots, s_k, x_1, \ldots, x_k)$ we consider two cases. If the event Forge occurs, then clearly $\mathcal{A}'$ outputs 1 with probability 1/2. In addition, if the event Forge does not occur, the simulated interaction is identical to the CCA2 interaction (a formal argument follows). Therefore, the probability that $\mathcal{A}'$ outputs 1 in this case is exactly $\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr\left[\mathsf{Forge}\right]$.

   To see that the above holds, note that the only difference between the CCA2 interaction and the simulated interaction is the distribution of the challenge ciphertext: In the CCA2 interaction the value vk in the challenge ciphertext is a randomly chosen verification key, and in the simulated interaction the value vk is chosen ahead of time by $\mathcal{A}$. As was argued in the proof of Claim 4.3, as long as the event Forge does not occur, the distribution of vk in the challenge ciphertext is identical in the two cases.

The above observations imply that

$$\left|\Pr\left[\mathcal{A}' \text{ outputs } 1 \mid w = h(s_1, \ldots, s_k, x_1, \ldots, x_k)\right] - \Pr\left[\mathcal{A}' \text{ outputs } 1 \mid w \text{ is random}\right]\right|$$
$$= \left|\Pr\left[\mathsf{Success} \wedge \overline{\mathsf{Forge}}\right] + \frac{1}{2}\Pr\left[\mathsf{Forge}\right] - \frac{1}{2}\right| \quad ,$$

A standard argument (see, for example, [22, Chapter 3.4]) can be applied to efficiently transform $\mathcal{A}'$ into a predictor $\mathcal{P}$ that predicts $h(s_1, \ldots, s_k, x)$ with the same probability. ■

■

6 A Black-Box Separation

In this section we show that there is no fully-black-box construction of lossy trapdoor functions (with even a single bit of lossiness) from injective trapdoor functions that are secure under correlated products. We show that this holds for the seemingly strongest form of correlated product, where independently chosen functions are evaluated on the same input (i.e., we consider the uniform k-repetition distribution).

Our proof consists of constructing an oracle $\mathcal{O}$ relative to which there exists a collection of injective trapdoor functions that are permutations secure under a correlated product[5], but there are no collections of lossy trapdoor functions. In what follows, we describe the oracle $\mathcal{O}$, and show that it breaks the security of any collection of lossy trapdoor functions.

The oracle. The oracle $\mathcal{O}$ is of the form $(\tau, \mathsf{Sam}^\tau)$, where τ is a collection of trapdoor permutations, and Sam^τ is an oracle that samples random collision. Specifically, Sam receives as input a description of a circuit C (which may contain τ-gates), chooses a random input w, and then samples a uniformly distributed $w' \in C^{-1}(C(w))$.

We now explain how exactly Sam samples w and w'. We provide Sam with a collection of permutations $\mathcal{F}$, where for every possible circuit C the collection $\mathcal{F}$ contains two permutations f_C^1 and f_C^2 over the domain of C. Given a circuit $C : \{0,1\}^m \to \{0,1\}^{\ell(m)}$, for some m and $\ell(m)$, the oracle Sam uses f_C^1 to compute $w = f_C^1(0^m)$. Then, it computes $w' = f_C^2(t)$ for the lexicographically smallest $t \in \{0,1\}^m$ such that $C(f_C^2(t)) = C(w)$. Note that whenever the permutations f_C^1 and f_C^2 are chosen uniformly at random, and independently of all other permutations in $\mathcal{F}$, then w is uniformly distributed over $\{0,1\}^m$, and w' is uniformly distributed over $C^{-1}(C(w))$. In the remainder of the proof, whenever we consider the probability of an event over the choice of the collection $\mathcal{F}$, we mean that for each circuit C, two permutations f_C^1 and f_C^2 are chosen uniformly at random and independently of all other permutations. A complete and formal description of the oracle is provided in Figure 1.

On input a circuit $C : \{0,1\}^m \to \{0,1\}^{\ell(m)}$, the oracle $\mathsf{Sam}^{\tau,\mathcal{F}}$ acts as follows:

1. Compute $w = f_C^1(0^m)$.
2. Compute $w' = f_C^2(t)$ for the lexicographically smallest $t \in \{0,1\}^m$ such that $C(f_C^2(t)) = C(w)$.
3. Output (w, w')

Figure 1: The oracle Sam.

Distinguishing between injective functions and lossy functions. The oracle Sam can be easily used to distinguish between the injective mode and the lossy mode of any collection of $(n,1)$-lossy functions. Consider the following distinguisher A: given a circuit C (which may contain τ-gates[6]), which is a description of either an injective function or a lossy function (with image size at most 2^{n-1}), A queries Sam with C.

If Sam returns (w, w') such that $w = w'$, then A outputs 1, and otherwise A outputs 0. Clearly, if C corresponds to an injective function, then always $w = w'$ and A outputs 1. In addition, if C

[5] These functions are in fact enhanced trapdoor permutations, but we note that this is not essential for our result.

[6] We allow the circuits given as input to Sam to contain τ-gates, but we do not allow them to contain Sam-gates. This suffices, however, for ruling out fully-black-box constructions (as in the work of Hsiao and Reyzin [33]).

corresponds to a lossy function, then with probability at least 1/4 it holds that $w \neq w'$, where the probability is taken over the randomness of Sam (i.e., over the collection $\mathcal{F}$)[7].

Outline of the proof. For simplicity we first consider only two permutations. Then, we extend our argument to more than two permutations, and to trapdoor permutations. Our goal is to upper bound the success probability of circuits having oracle access to Sam in the task of inverting $(\pi_1(x), \pi_2(x))$ for random permutations $\pi_1, \pi_2 \in \Pi_n$ and a random $x \in \{0,1\}^n$ (where Π_n is the set of all permutations over $\{0,1\}^n$). We prove the following theorem:

Theorem 6.1. *For any circuit A of size at most $2^{n/40}$ and for all sufficiently large n, it holds that*

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \right] \leq \frac{1}{2^{n/40}} \ .$$

Consider a circuit A which is given as input $(\pi_1(x), \pi_2(x))$, and whose goal is to retrieve x. The idea underlying the proof is to distinguish between two cases: one in which A obtains information on x via one of its Sam-queries, and the other in which none of A's Sam-queries provides information on x. More specifically, we define:

Definition 6.2. *A Sam-query C produces a x-*hit *if Sam outputs (w, w') such that some π_1-gate or π_2-gate in the computations of $C(w)$ or $C(w')$ has input x.*

Given π_1, π_2, $\mathcal{F}$, a circuit A, and a pair $(\pi_1(x), \pi_2(x))$, we denote by SamHIT_x the event in which one of the Sam-queries made by A produces a x-hit. From this point on, the proof proceeds in two modular parts. In the first part of the proof, we consider the case that the event SamHIT_x does not occur, and generalize an argument of Haitner et al. [28] (who in turn generalized the reconstruction lemma of Gennaro and Trevisan [18]). We show that if a circuit A manages to invert $(\pi_1(x), \pi_2(x))$ for many x's, then π_1 and π_2 have a short representation given A. This enables us to prove the following lemma:

Lemma 6.3. *For any circuit A of size at most $2^{n/7}$ and for all sufficiently large n, it holds that*

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}_x} \right] \leq 2^{-n/8} \ .$$

In the second part of the proof, we show that the case where the event SamHIT_x does occur can be reduced to the case where the event SamHIT_x does not occur. Given a circuit A that tries to invert $(\pi_1(x), \pi_2(x))$, we construct a circuit M that succeeds almost as well as A, without M's Sam-queries producing any x-hits. This proof is a simpler case of a similar argument due to Haitner et al. [28]. The following theorem is proved:

Lemma 6.4. *For any circuit A of size $s(n)$, if*

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}((\pi_1(x), \pi_2(x))) = x \right] \geq \frac{1}{s(n)}$$

for infinitely many values of n, then there exists a circuit M of size $O(s(n))$ such that

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}((\pi_1(x), \pi_2(x))) = x \ \wedge \ \overline{\mathsf{SamHIT}_x} \right] \geq \frac{1}{s(n)^5}$$

for infinitely many values of n.

[7] Note that with probability at least 1/2 over the choice of w it holds that $|C^{-1}(C(w))| > 1$. In this case, when sampling $w' \in C^{-1}(C(w))$ it holds that $w' \neq w$ with probability at least 1/2. Therefore, if C corresponds to a lossy function, then with probability at least 1/4 it holds that $w \neq w'$. This is a rather pessimistic analysis, but it is clearly sufficient for our purposes.

In what follows we show that Theorem 6.1 in obtained as a straightforward corollary of Lemmata 6.3 and 6.4. In Section 6.1 we prove Lemma 6.3, and in Section 6.2 we prove Lemma 6.4. Finally, in Section 6.3 we extend Theorem 6.1 to consider more than two permutations and to consider trapdoor permutations.

Proof of Theorem 6.1. Assume towards a contradiction that there exists a circuit A of size at most $s(n) = 2^{n/40}$ such that

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \right] \geq \frac{1}{2^{n/40}} \ ,$$

for infinitely many values of n. Lemma 6.4 states that there exists a circuit M of size $O(s(n)) \leq 2^{n/7}$ such that

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}((\pi_1(x), \pi_2(x))) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x \right] \geq \frac{1}{s(n)^5} = \frac{1}{2^{n/8}}$$

for infinitely many values of n. This, however, contradicts Lemma 6.3.

6.1 The Reconstruction Lemma

In this section we prove Lemma 6.3. The idea underlying the reconstruction argument is the following: Fix any two permutations π_1 and π_2. If a circuit A manages to invert $(\pi_1(x), \pi_2(x))$ on some set of x's, then given the circuit A, the permutations π_1 and π_2 can be described without specifying their value on a relatively large fraction of this set.

Claim 6.5. *For every $\pi_1, \pi_2 \in \Pi_n$, $\mathcal{F}$, circuit A of size s and integer n, if*

$$\Pr_{x \leftarrow \{0,1\}^n} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x \right] \geq \epsilon \ ,$$

then, given $\mathcal{F}$ and A, the permutations π_1 and π_2 can be described using $\log \binom{2^n}{a} + \log \binom{2^{2n}}{a} + 2\log((2^n - a)!)$ *bits, where $a \geq \epsilon 2^n / \left(2s^2\right)$.*

Proof. Denote by $I \subseteq \{0,1\}^n$ the set of points $x \in \{0,1\}^n$ on which A inverts $(\pi_1(x), \pi_2(x))$ with no x-hits. We claim that there exists a relatively large set $X \subseteq I$, such that π_1 and π_2 are completely determined by $\mathcal{F}$, A, X, $Y = \{(\pi_1(x), \pi_2(x)) : x \in X\}$, and the values of π_1 and π_2 on $\{0,1\}^n \setminus X$.

We define the set X via the following sequential process. Let $P = \{(\pi_1(x), \pi_2(x)) : x \in I\}$. Initially X is empty, and we remove the lexicographically smallest element $(y_1, y_2) = (\pi_1(x), \pi_2(x))$ from P and insert x into X. Then, we follow the computation $A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(y_1, y_2)$, denote by $C_1, \ldots, C_q$ the circuits on which A queries Sam, and by $(w_1, w'_1), \ldots, (w_q, w'_q)$ the corresponding answers. In addition, denote by $x_1, \ldots, x_t$ the inputs of all the π_1-gates and π_2-gates in the computations of $C_1(w_1), C_1(w'_1), \ldots, C_q(w_q), C_q(w'_q)$ and the inputs of all A's direct queries to π_1 and to π_2. We now remove $(\pi_1(x_1), \pi_2(x_1)), \ldots, (\pi_1(x_q), \pi_2(x_q))$ from the set P (note that these are not necessarily in the set P). Then, remove the lexicographically smallest element from the remaining elements of P, and continue in the same manner until the set P is emptied.

Note that at each iteration one element is inserted into the set X, and at most $s^2 + s + 1 \leq 2s^2$ elements are removed from the set P (the number q of Sam-queries made by A is at most s, and in each circuit given by A as input to Sam the number of π_1-gates and π_2-gates is again at most s. In addition, A may directly query π_1 and π_2 on at most s inputs). Since the set P initially contains at least $\epsilon 2^n$ elements, then when the process terminates we have that $|X| \geq \epsilon 2^n/(2s^2)$.

We now claim that π_1 and π_2 are completely determined by $\mathcal{F}$, A, X, $Y = \{(\pi_1(x), \pi_2(x)) : x \in X\}$, and the values of π_1 and π_2 on $\{0,1\}^n \setminus X$. More specifically, we show that the values $\{(\pi_1(x), \pi_2(x)) : x \in X\}$ can be reconstructed. For each $(y_1, y_2) \in Y$ taken in lexicographical increasing order, we reconstruct x^* such that $(y_1, y_2) = (\pi_1(x^*), \pi_2(x^*))$ by simulating $(\pi_1, \pi_2$ and $\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}$ in the computation $A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(y_1, y_2)$. Note that if the simulation is correct, then A will output x^*. On input a query C_i with two corresponding permutation $f^1_{C_i}, f^2_{C_i} \in \mathcal{F}$, we simulate Sam as follows:

1. Let $w_i = f^1_{C_i}(0^m)$.

2. Compute $C(w_i)$. It is not immediately clear that we can indeed compute this value without full access to π_1 and π_2 since the circuit C may contain π_1-gates and π_2-gates. We need to show that we can answer of the π_1-queries and π_2-queries in this computation $C(w_i)$. This computation may involve four possible π_1-queries (an identical argument holds for π_2-queries):

   - π_1-query on $x \in \{0,1\}^n \setminus X$. The value is explicitly given.
   - π_1-query on $x \in X$ for which $(\pi_1(x), \pi_2(x)) <_{lex} (y_1, y_2)$. The required value was already reconstructed.
   - π_1-query on $x \in X$ for which $(\pi_1(x), \pi_2(x)) >_{lex} (y_1, y_2)$. We claim that this is impossible. Assume towards a contradiction that such a π_1-query is made. This implies that both $x \in X$ and $x^* \in X$ (recall that x^* is such that $(y_1, y_2) = (\pi_1(x^*), \pi_2(x^*))$). Consider the process which defined the set X. At the beginning of this process we had that both $(\pi_1(x), \pi_2(x)) \in P$ and $(\pi_1(x^*), \pi_2(x^*)) \in P$, and in each iteration we chose the minimal element from the remaining elements in P. Since $(\pi_1(x), \pi_2(x)) >_{lex} (\pi_1(x^*), \pi_2(x^*))$, then we chose $(\pi_1(x^*), \pi_2(x^*))$ before $(\pi_1(x), \pi_2(x))$. This implies, however, that we then removed $(\pi_1(x), \pi_2(x))$ from P (since a π_1-query on x is made in the computation of $C(w_i)$). Thus, it is not possible that $x \in X$.
   - π_1-query on $x \in X$ for which $(\pi_1(x), \pi_2(x)) = (y_1, y_2)$. Impossible, otherwise the Sam-query C_i produces a x-hit.

3. Let $w'_i = f^2_{C_i}(t)$ for the minimal t such that $C_i(f^2_{C_i}(t))$ can be computed (i.e., all π_1-queries and π_2-queries can be answered) and its resulting value is $C_i(w_i)$. This value can be computed for the same reason that $C_i(w_i)$ can be computed.

4. Output (w_i, w'_i).

We also have to show that we can answer all of A's direct π_1-queries (an identical argument holds for π_2-queries). Whenever A asks for the value of π_1 on some value x, we act as follows: if this value is already known (i.e., explicitly given or already reconstructed), then we output $\pi_1(x)$ to A. Otherwise, if the value is not known, we claim that it must be that $x = x^*$, and in this case we have successfully reconstructed the desired value and halt. Indeed, there are four possible such queries:

- π_1-query on $x \in \{0,1\}^n \setminus X$. The value is explicitly given.
- π_1-query on $x \in X$ for which $(\pi_1(x), \pi_2(x)) <_{lex} (y_1, y_2)$. The required value was already reconstructed.
- π_1-query on $x \in X$ for which $(\pi_1(x), \pi_2(x)) >_{lex} (y_1, y_2)$. This is impossible (as above).
- π_1-query on $x \in X$ for which $(\pi_1(x), \pi_2(x)) = (y_1, y_2)$. In this case $x = x^*$.

Thus, we can successfully reconstruct the values of π_1 ad π_2 on the set X. Finally, note that describing the sets X and Y, and the values of π_1 and π_2 on the set $\{0,1\}^n \setminus X$ requires $\log \binom{2^n}{|X|} + \log \binom{2^{2n}}{|X|} + 2\log((2^n - |X|)!)$ bits. ■

Now we are able to prove the following lemma, which is a stronger form of Lemma 6.3.

Lemma 6.6. *For every $\mathcal{F}$, circuit A of size at most $2^{n/7}$ and for all sufficiently large n,*

$$\Pr_{\substack{\pi_1,\pi_2 \leftarrow \Pi_n \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}_x} \right] \leq 2^{-n/8} \ .$$

Proof. Let $\epsilon = 2^{-n/7}$, then Claim 6.5 implies that for every circuit A of size $s \leq 2^{n/7}$ and for every collection $\mathcal{F}$ of permutations, the *fraction* of pairs of permutations $(\pi_1, \pi_2) \in \Pi_n \times \Pi_n$ for which

$$\Pr_{x \leftarrow \{0,1\}^n} \left[A^{\pi_1,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}_x} \right] \geq 2^{-n/7}$$

is at most

$$\frac{\binom{N}{a}\binom{N^2}{a}((N-a)!)^2}{(N!)^2} \ ,$$

where $N = 2^n$, and $a \geq 2^{-n/7} \cdot N/(2s^2) \geq N^{4/7}/2$. Using the inequalities $a! \geq (a/e)^a$ and $(x/y)^y \leq \binom{x}{y} \leq (xe/y)^y$, we can bound the above expression as follows

$$\begin{aligned}
\frac{\binom{N}{a}\binom{N^2}{a}((N-a)!)^2}{(N!)^2} &= \frac{\binom{N^2}{a}}{\binom{N}{a}(a!)^2} \\
&\leq \frac{\left(\frac{N^2 e}{a}\right)^a}{\left(\frac{N}{a}\right)^a \left(\frac{a}{e}\right)^{2a}} \\
&= \left(\frac{N \cdot e^3}{a^2}\right)^a \\
&\leq \left(\frac{4 \cdot e^3}{N^{1/7}}\right)^a \\
&\leq \left(\frac{1}{2}\right)^{N^{4/7}/2} \ ,
\end{aligned}$$

for sufficiently large N. Therefore,

$$\Pr_{\substack{\pi_1,\pi_2 \leftarrow \Pi_n \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(y) = x \ \wedge \ \overline{\mathsf{SamHIT}_y} \right] \leq 2^{-N^{4/7}/2} + 2^{-n/7} \leq 2^{-n/8} \ .$$

■

6.2 Avoiding x-Hits

In this section we prove Lemma 6.4. Given a circuit A of size $s(n)$ such that

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \right] \geq \frac{1}{s(n)} \ ,$$

we would like to construct a circuit M which is almost as successful as A, but its Sam-queries do not produce any x-hits. Recall (Definition 6.2), that we say that a Sam-query C produces a x-hit if Sam outputs (w, w') such that some π_1-gate or π_2-gate in the computations of $C(w)$ or $C(w')$ has input x. In addition, we denoted by SamHIT_x the event in which at least one Sam-query produces a x-hit.

Description of M. On input (y_1, y_2), the circuit M feeds A with (y_1, y_2) as its input, and delivers all of A's queries to Sam and to π_1 and π_2 with the following exception: for each Sam-query $C : \{0,1\}^m \to \{0,1\}^{\ell(m)}$ that A submits, M first chooses a random $z \in \{0,1\}^m$ and computes $C(z)$. If some π_1-gate or π_2-gate in the computation of $C(z)$ has input x such that $(y_1, y_2) = (\pi_1(x), \pi_2(x))$ then M outputs x and halts. Otherwise, M submits C to Sam and delivers the result (w, w') back to A. If M did not halt before the termination of A's computation, then it outputs the output of A and halts.

Proof of Lemma 6.4. The circuit M does not make any additional Sam-queries other than those made by A. Therefore, if A inverts $(\pi_1(x), \pi_2(x))$ without producing any x-hits in its Sam-queries, then so does M. Formally, if

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x \right] \geq \frac{1}{2s(n)} \ ,$$

then

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x \right] \geq \frac{1}{2s(n)} \ .$$

Thus, for the rest of the proof we focus on the more interesting case, in which A does produce an x-hit. That is, we assume that

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \mathsf{SamHIT}_x \right] \geq \frac{1}{2s(n)} \ . \tag{6.1}$$

We now fix π_1, π_2 and $x \in \{0,1\}^n$, and prove the following lemma:

Lemma 6.7. *For every π_1, π_2 and $x \in \{0,1\}^n$, if*

$$\Pr_{\mathcal{F}} \left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \mathsf{SamHIT}_x \right] \geq \frac{1}{8s(n)} \ , \tag{6.2}$$

then

$$\Pr_{\mathcal{F}} \left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x \right] \geq \frac{1}{1024 s(n)^3} \ .$$

Proof of Lemma 6.7. Fix π_1, π_2 and $x \in \{0,1\}^n$, and let $s = s(n)$. We introduce the following conventions and notations:

- Without loss of generality, the circuit A does not query π_1 or π_2 directly[8].
- We denote by $C_1, \dots, C_q$ the random variables corresponding to A's Sam-queries. In addition, we denote by $(w_1, w_1'), \dots, (w_q, w_q')$ the random variables corresponding to the answers returned by Sam.
- Given a circuit C and an input w, we say that w *produces a (C, x)-hit* if some π_1-gate or π_2-gate in the computation of $C(w)$ has input x.
- For every $1 \leq i \leq q$ we denote by α_i the probability that w_i produces a (C_i, x)-hit (note that this is exactly the same probability that w_i' produces a (C_i, x)-hit). Formally,

$$\alpha_i = \Pr_{w_i} \left[w_i \text{ produces a } (C_i, x)\text{-hit} \right] \ .$$

[8] The oracle Sam can be modified to output $(w, w', C(w))$, and therefore any π_1-query and π_2-query can be replaced by a single Sam-query by creating a circuit C that ignores its input and always outputs $\pi_1(x)$ or $\pi_2(x)$ for some x.

- For every $1 \leq i \leq q$, we denote by JUMP_i the event that $\alpha_i > 1/(32s^2)$, and let $\mathsf{JUMP} = \bigcup_{i=1}^{q} \mathsf{JUMP}_i$.

Equation 6.2 states that A has a noticeable probability in producing a x-hit. Notice that in this case the event JUMP must occur with noticeable probability. If JUMP does not occur, then the α_i's are too small in order to produce a x-hit with noticeable probability.

Claim 6.8. $\Pr_{\mathcal{F}}\left[\mathsf{SamHIT}_x \;\middle|\; \overline{\mathsf{JUMP}}\right] \leq 1/(16s)$.

Proof. Assuming that the event JUMP does not occur, that is $\alpha_i \leq 1/(32s^2)$ for every $1 \leq i \leq q$, it holds that

$$\begin{aligned}\Pr_{\mathcal{F}}\left[\mathsf{SamHIT}_x \;\middle|\; \overline{\mathsf{JUMP}}\right] &\leq \sum_{i=1}^{q} \Pr_{\mathcal{F}}\left[w_i \text{ or } w'_i \text{ produce a } (C_i, x)\text{-hit}\right] \\ &\leq \sum_{i=1}^{q} 2\alpha_i \leq s \cdot \frac{1}{16s^2} = \frac{1}{16s} \ .\end{aligned}$$

■

As a result of the previous claim, we now show that the event JUMP has noticeable probability.

Claim 6.9. $\Pr_{\mathcal{F}}[\mathsf{JUMP}] \geq 1/(16s)$.

Proof. On one hand, Equation 6.2 implies in particular that

$$\Pr_{\mathcal{F}}[\mathsf{SamHIT}_x] \geq \frac{1}{8s} \ .$$

However, on the other hand, Claim 6.8 implies that

$$\begin{aligned}\Pr_{\mathcal{F}}[\mathsf{SamHIT}_x] &\leq \Pr_{\mathcal{F}}[\mathsf{JUMP}] + \Pr_{\mathcal{F}}\left[\mathsf{SamHIT}_x \;\middle|\; \overline{\mathsf{JUMP}}\right] \\ &\leq \Pr_{\mathcal{F}}[\mathsf{JUMP}] + \frac{1}{16s} \ .\end{aligned}$$

Therefore,

$$\Pr_{\mathcal{F}}[\mathsf{JUMP}] \geq \frac{1}{8s} - \frac{1}{16s} \geq \frac{1}{16s} \ .$$

■

Assume now that the event JUMP occurs, and denote by i^* the minimal $1 \leq i \leq q$ for which JUMP_i occurs. When A submits the query C_{i^*}, then M has probability $\alpha_{i^*} > 1/(32s^2)$ to retrieve x without submitting the query to Sam. In addition, since i^* is the minimal $1 \leq i \leq q$ for which JUMP_i occurs, then with high probability Sam's answers to $C_1, \ldots, C_{i^*-1}$ do not produce an x-hit. The following claim concludes the proof of Lemma 6.7.

Claim 6.10. $\Pr_{\mathcal{F}}\left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x), \pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}_x}\right] \geq 1/\left(1024s^3\right)$.

Proof. Given that the event JUMP occurs, denote by i^* the minimal $1 \leq i \leq q$ for which JUMP_i occurs. Whenever the event JUMP occurs, we consider the following events:

- None of the queries $C_1, \ldots, C_{i^*-1}$ produces a x-hit. Since for every such query C_i the event JUMP_i does not occur, then, exactly as in the proof of Claim 6.8, the probability of this event is at least $1 - 1/(16s)$.

- Given C_{i^*}, M samples a random z which produces a (C_{i^*}, x)-hit. Since JUMP_{i^*} occurs, the probability of this event is $\alpha_{i^*} \geq 1/(32s^2)$.

Note that these two events are independent (since the permutations in $\mathcal{F}$ are chosen independently). Putting these together, we obtain

$$\begin{aligned}\Pr_{\mathcal{F}}\left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x),\pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x\right] &\geq \Pr_{\mathcal{F}}[\mathsf{JUMP}] \cdot \left(1 - \frac{1}{16s}\right) \cdot \frac{1}{32s^2} \\ &\geq \frac{1}{16s} \cdot \frac{1}{2} \cdot \frac{1}{32s^2} \\ &\geq \frac{1}{1024s^3} \quad .\end{aligned}$$

■

This concludes the proof of Lemma 6.7. We now turn to complete the proof of Lemma 6.4 using a standard averaging argument. Recall that we were left to deal with the case that

$$\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}}\left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x),\pi_2(x)) = x \ \wedge \ \mathsf{SamHIT}_x\right] \geq \frac{1}{2s(n)} \quad .$$

Let

$$T = \left\{(x,\pi_1,\pi_2) : \Pr_{\mathcal{F}}\left[A^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x),\pi_2(x)) = x \ \wedge \ \mathsf{SamHIT}_x\right] \geq \frac{1}{8s(n)}\right\} \quad .$$

Then $\Pr_{x,\pi_1,\pi_2}[(x,\pi_1,\pi_2) \in T] \geq 1/8s(n)$, and Lemma 6.7 implies that for every such $(x,\pi_1,\pi_2) \in T$ we have

$$\Pr_{\mathcal{F}}\left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}(\pi_1(x),\pi_2(x)) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x\right] \geq \frac{1}{1024s(n)^3} \quad .$$

Therefore

$$\begin{aligned}&\Pr_{\substack{\pi_1,\pi_2,\mathcal{F} \\ x \leftarrow \{0,1\}^n}}\left[M^{\pi_1,\pi_2,\mathsf{Sam}^{\pi_1,\pi_2,\mathcal{F}}}((\pi_1(x),\pi_2(x))) = x \ \wedge \ \overline{\mathsf{SamHIT}}_x\right] \\ &\geq \Pr_{\substack{\pi_1,\pi_2 \\ x \leftarrow \{0,1\}^n}}[(x,\pi_1,\pi_2) \in T] \cdot \frac{1}{1024s(n)^3} \\ &\geq \frac{1}{8s(n)} \cdot \frac{1}{1024s(n)^3} \\ &\geq \frac{1}{s(n)^5} \quad .\end{aligned}$$

6.3 Extensions of Theorem 6.1

In this section we extend Theorem 6.1 to consider more than two permutations and to consider trapdoor permutations.

More than two permutations. The proof for the case of $k > 2$ permutations is obtained as a direct generalization. Recall that the proof consists of two parts: the first part proves the reconstruction lemma, and the second part shows that x-hits can be avoided. We note that the second part of the proof is oblivious to the number of permutations, and therefore the proof of

Lemma 6.4 remains exactly the same. The first part of the proof is not oblivious to the number of permutations, but can be easily adapted as follows.

The exact same proof of Claim 6.5 easily generalizes to consider permutations $\pi_1, \ldots, \pi_k$. In this case, given $\mathcal{F}$ and A, the permutations $\pi_1, \ldots, \pi_k$ can be described using $\log \binom{2^n}{a} + \log \binom{2^{kn}}{a} + k \log((2^n - a)!)$ bits, where $a \geq \epsilon 2^n / (2s^2)$. Then, in the proof of Lemma 6.6 the fraction of permutations $\pi_1, \ldots, \pi_k$ on which A successfully inverts is at most

$$\frac{\binom{N}{a}\binom{N^k}{a}((N-a)!)^k}{(N!)^k} \leq \left(\frac{4e^{k+1}}{N^{1/7}}\right)^a ,$$

where $N = 2^n$, and $a \geq 2^{-n/7} \cdot N/(2s^2) \geq N^{4/7}/2$. As long as $k \leq cn$, for some constant $0 < c < 1$, then this fraction is at most $2^{-n/7}$, and the exact same argument goes through.

Trapdoor permutations. The extension to trapdoor permutations is almost identical to the corresponding extension of Haitner et al. [28], and therefore we only provide here the intuition. The basic idea in extending the result for trapdoor permutation is in applying Theorem 6.1 twice. Consider a collection $\tau = (G, F, F^{-1})$ of trapdoor permutations over $\{0,1\}^n$, and let A be a circuit which successfully inverts the correlated product of two independently chosen trapdoor permutations. That is, we independently sample two pairs $(pk_1, td_1) \leftarrow G(1^n)$ and $(pk_2, td_2) \leftarrow G(1^n)$, sample a uniformly distributed $x \in \{0,1\}^n$, and A given input $(F(pk_1, x), F(pk_2, x))$ outputs x.

We consider now two cases. In the first case, during A's computation the procedure F^{-1} is queried with either td_1 or td_2. Without loss of generality assume that F^{-1} is queried with td_1. In this case the circuit A can be used to invert a random permutation $\pi = G$ on a random input td_1. In the second case, the procedure F^{-1} is not queried with wither one of td_1 and td_2. In this case the circuit A can be used to invert the correlated product $(\pi_1(x), \pi_2(x))$, where $\pi_1 = F(pk_1, \cdot)$ and $\pi_2 = F(pk_2, \cdot)$.

Acknowledgments

We thank Oded Goldreich, Moni Naor, Chris Peikert, and Omer Reingold for useful discussions. In particular, we thank Oded for suggesting the relaxation that led to the generalized scheme in Section 5. We also thank the anonymous SICOMP reviewers for their comments.

References

[1] A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *Advances in Cryptology – CRYPTO '08*, pages 335–359, 2008.

[2] D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices of the American Mathematical Society*, 46(2):203–213, 1999.

[3] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.

[4] D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In *Topics in Cryptology - CT-RSA '05, The Cryptographers' Track at the RSA Conference*, pages 87–103, 2005.

[5] X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In *Proceedings of the 12th ACM Conference on Computer and Communications Security*, pages 320–329, 2005.

[6] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *Advances in Cryptology - EUROCRYPT '04*, pages 207–222, 2004.

[7] D. Cash, E. Kiltz, and V. Shoup. The Twin Diffie-Hellman problem and applications. In *Advances in Cryptology - EUROCRYPT '08*, pages 127–145, 2008.

[8] S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *Proceedings of the 5th Theory of Cryptography Conference*, pages 427–444, 2008.

[9] R. Cramer, G. Hanaoka, D. Hofheinz, H. Imai, E. Kiltz, R. Pass, A. Shelat, and V. Vaikuntanathan. Bounded CCA2-secure encryption. In *Advances in Cryptology - ASIACRYPT '07*, pages 502–518, 2007.

[10] R. Cramer, D. Hofheinz, and E. Kiltz. A twist on the Naor-Yung paradigm and its application to efficient CCA-secure encryption from hard search problems. In *Proceedings of the 7th Theory of Cryptography Conference*, pages 146–164, 2010.

[11] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '98*, pages 13–25, 1998.

[12] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen-ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[13] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[14] R. Dowsley, J. Müller-Quade, and A. C. A. Nascimento. A CCA2 secure public-key encryption scheme based on the McEliece assumptions in the standard model. In *Topics in Cryptology – CT-RSA 2009*, pages 240–251, 2009.

[15] E. Elkind and A. Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive, Report 2002/042, 2002.

[16] D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. To appear in *Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography*, 2010.

[17] R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM Journal on Computing*, 35(1):217–246, 2005.

[18] R. Gennaro and L. Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 305–313, 2000.

[19] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 325–335, 2000.

[20] Y. Gertner, T. Malkin, and S. Myers. Towards a separation of semantic and CCA security for public key encryption. In *Proceedings of the 4th Theory of Cryptography Conference*, pages 434–455, 2007.

[21] Y. Gertner, T. Malkin, and O. Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 126–135, 2001.

[22] O. Goldreich. *Foundations of Cryptography – Volume 1: Basic Tools*. Cambridge University Press, 2001.

[23] O. Goldreich. *Foundations of Cryptography – Volume 2: Basic Applications*. Cambridge University Press, 2004.

[24] O. Goldreich, R. Impagliazzo, L. A. Levin, R. Venkatesan, and D. Zuckerman. Security preserving amplification of hardness. In *Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 318–326, 1990.

[25] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, 1989.

[26] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[27] S. Goldwasser and V. Vaikuntanathan. New constructions of correlation-secure trapdoor functions and CCA-secure encryption schemes. Manuscript, 2008.

[28] I. Haitner, J. J. Hoch, O. Reingold, and G. Segev. Finding collisions in interactive protocols – A tight lower bound on the round complexity of statistically-hiding commitments. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 669–679, 2007.

[29] G. Hanaoka and K. Kurosawa. Efficient chosen ciphertext secure public key encryption under the computational Diffie-Hellman assumption. In *Advances in Cryptology - ASIACRYPT '08*, pages 308–325, 2008.

[30] J. Håstad. Solving simultaneous modular equations of low degree. *SIAM Journal on Computing*, 17(2):336–341, 1988.

[31] B. Hemenway, S. Lu, and R. Ostrovsky. Correlated product security from any one-way function and the new notion of decisional correlated product security. Cryptology ePrint Archive, Report 2010/100, 2010.

[32] D. Hofheinz and E. Kiltz. Practical chosen ciphertext secure encryption from factoring. In *Advances in Cryptology - EUROCRYPT '09*, pages 313–332, 2009.

[33] C.-Y. Hsiao and L. Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Advances in Cryptology - CRYPTO '04*, pages 92–105, 2004.

[34] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 44–61, 1989.

[35] E. Kiltz, K. Pietrzak, M. Stam, and M. Yung. A new randomness extraction paradigm for hybrid encryption. In *Advances in Cryptology - EUROCRYPT '09*, pages 590–609, 2009.

[36] J. H. Kim, D. R. Simon, and P. Tetali. Limits on the efficiency of one-way permutation-based hash functions. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 535–542, 1999.

[37] Y. Lindell. A simpler construction of CCA2-secure public-key encryption under general assumptions. *Journal of Cryptology*, 19(3):359–377, 2006.

[38] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN Prog. Rep., Jet Prop. Lab.*, pages 114–116, 1978.

[39] P. Mol and S. Yilek. Chosen-ciphertext security from slightly lossy trapdoor functions. To appear in *Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography*, 2010.

[40] S. Myers and abhi shelat. Bit encryption is complete. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009.

[41] M. Naor and O. Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *Journal of Computer and System Sciences*, 58(2):336–375, 1999.

[42] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, 1990.

[43] R. Pass, A. Shelat, and V. Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *Advances in Cryptology - CRYPTO '06*, pages 271–289, 2006.

[44] R. Pass, A. Shelat, and V. Vaikuntanathan. Relations among notions of non-malleability for encryption. In *Advances in Cryptology - ASIACRYPT '07*, pages 519–535, 2007.

[45] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. Cryptology ePrint Archive, Report 2008/481, 2008.

[46] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 187–196, 2008.

[47] C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '91*, pages 433–444, 1991.

[48] O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of reducibility between cryptographic primitives. In *Proceedings of the 1st Theory of Cryptography Conference*, pages 1–20, 2004.

[49] A. Rosen and G. Segev. Efficient lossy trapdoor functions based on the composite residuosity assumption. Cryptology ePrint Archive, Report 2008/134, 2008.

[50] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 543–553, 1999.

[51] D. R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology - EUROCRYPT '98*, pages 334–345, 1998.

[52] Y. Vahlis. Two is a crowd? A black-box separation of one-wayness and security under correlated inputs. In *Proceedings of the 7th Theory of Cryptography Conference*, pages 165–182, 2010.

[53] A. C. Yao. Theory and applications of trapdoor functions. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982.