STRONGER SECURITY NOTIONS FOR TRAPDOOR FUNCTIONS AND APPLICATIONS

A Thesis
Presented to
The Academic Faculty

by

Adam O'Neill

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
December 2010

STRONGER SECURITY NOTIONS FOR TRAPDOOR FUNCTIONS AND APPLICATIONS

To my parents,

John (Chuck) and Phyllis O'Neill,

and my sister Katie,

for their unconditional support.

ACKNOWLEDGEMENTS

My Ph.D. studies have been a significant undertaking, which would not have been possible without the help and guidance of many people (please forgive any omissions).

First of all, I'd like to thank my parents, for encouraging me to pursue higher education and giving me the opportunity to do so.

My intellectual development was greatly fostered during my time as an undergraduate at UCSD, and for that I have many friends and teaching assistants to thank. I would especially like to thank Daniel Bryant for helping me during my freshman year, and Derek Newland for inviting me to do an independent study with him. I would also like to thank Mihir Bellare for taking the time to help undergraduates find out about graduate school and encouraging them to take graduate-level courses.

In graduate school, Sasha Boldyreva has been a lot of fun to have as an advisor, and I thank her for believing in me and all the effort she has put into helping me become a better researcher. Early on, I also received helpful advice from Subhash Khot and Dana Randall. During the course of my studies, I also benefited tremendously from interaction with other members of the cryptographic community, especially Michel Abdalla, Mihir Bellare, Yevgeniy Dodis, Serge Fehr, Marc Fischlin, Craig Gentry, Vipul Goyal, Eike Kiltz, Dick Lipton, Payman Mohassel, Chris Peikert, Krzysztof Pietrzak, Adam Smith, and Ramarathnam Venkatesan. Additionally, I would like to thank my fellow students for helping me not go (completely) crazy during this time, especially Gagan Goel and Ashish Sangwan.

Lastly, thanks to my landlords Bruce, Kim, and Hannah for making my stay in Atlanta pleasant.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

Trapdoor functions, introduced in the seminal paper of Diffie and Hellman [34], are a fundamental notion in modern cryptography. Informally, trapdoor functions are (injective) functions that are easy to evaluate but hard to invert unless given an additional input called the trapdoor. Specifically, the classical security notion considered for trapdoor functions is *one-wayness*, which asks that it be hard to invert (except with very small probability) a uniformly random point in the range without the trapdoor.

Motivated by the demands of emerging applications of cryptography as well as stronger security properties desired from higher-level cryptographic primitives constructed out of trapdoor functions, this thesis studies new strengthenings to the classical notion of one-way trapdoor functions and their applications. Our results are organized along two separate threads, wherein we introduce two new cryptographic primitives that strengthen the notion of one-wayness for trapdoor functions in different ways:

Deterministic Encryption: Our notion of deterministic (public-key) encryption addresses the weaknesses of using trapdoor functions directly for encryption articulated by Goldwasser and Micali [47], to the extent possible *without* randomizing the encryption function (whereas Goldwasser and Micali address them using randomized encryption). Specifically, deterministic encryption ensures no partial information is leaked about a high-entropy plaintext or even multiple correlated such plaintexts. Deterministic encryption has applications to fast search on encrypted data, securing legacy protocols, and "hedging" randomized encryption against bad randomness (cf. [6]). We design a conceptually appealing semantic-security style definition of security for deterministic encryption as well as an easier-to-work-with but equivalent indistinguishability style definition. In the random oracle model of Bellare and Rogaway [11], we show a secure construction of deterministic encryption for an unbounded number of arbitrarily correlated high-entropy plaintexts based on any randomized encryption scheme, as well as length-preserving such construction based on RSA. In the standard model, we develop a general framework for constructing deterministic encryption schemes based on a new notion of "robust" hardcore functions. We show a secure construction of deterministic for a single high-entropy plaintext based on exponentially-hard one-way trapdoor functions; single-message security is equivalent to security for an unbounded number of messages drawn from a *block-source* (where each subsequent message has high entropy conditioned on the previous) by a result of Fehr [41]. We also show a secure construction of deterministic encryption for a *bounded* number of arbitrarily correlated high-entropy plaintexts (or an unbounded number of messages drawn from a *q-block-source*, where the "blocks" consists of q messages and within each block are arbitrarily correlated high-entropy plaintexts) based on the notion of lossy trapdoor functions introduced by Peikert and Waters [68].

Adaptive Trapdoor Functions: Our notion of adaptive trapdoor functions asks that one-wayness be preserved in the presence of an inversion oracle that can be queried on some range points. The main application we give is the construction of black-box chosen-ciphertext secure public-key encryption from weaker general assumptions. ("Black-box" means that the specific code implementing the trapdoor function is not used in the construction, which typically incurs a huge efficiency cost.) Namely, we show such a construction of chosen-ciphertext secure public-key encryption from adaptive trapdoor functions. We then show that adaptive trapdoor functions can be realized from the recently introduced notions of lossy trapdoor functions by Peikert and Waters [68] and correlated-product secure trapdoor functions by Rosen and Segev [72]. In fact, by extending a recent result of Vahlis [76] we show adaptivity is strictly *weaker* than the latter notions (in a black-box sense). As a consequence, adaptivity is the weakest security property of trapdoor functions known to imply black-box chosen-ciphertext security. Additionally, by slightly extending our framework and considering "tag-based" adaptive trapdoor functions, we obtain exactly the chosen-ciphertext secure encryption schemes proposed in [68, 72], thereby unifying them, although the schemes we obtain via adaptive trapdoor functions are actually more efficient. Finally, we show that adaptive trapdoor functions can be realized from a (non-standard) computational assumption on RSA inversion, leading to a very efficient RSA-based chosen-ciphertext secure encryption scheme in the standard model.

CHAPTER I

INTRODUCTION

A central thrust of modern cryptography is studying stronger security notions for cryptographic primitives, providing definitions, constructions, and applications. Typically, this is done for "higher-level" cryptographic primitives, such as encryption or digital signature schemes. In this thesis, we instead pursue such a thrust for a "lower-level" cryptographic primitive, namely trapdoor functions. Here the distinction between "higher-level" and "lower-level" primitives is that the latter are usually simpler and the former construted out of the latter. We find that a study of stronger security notions for trapdoor functions is both timely from a practical perspective, as it provides notions that are useful in emerging applications and computing environments, as well as foundationally interesting, raising natural theoretical questions which, somewhat surprisingly, have not been studied before.

1.1 Background on Trapdoor Functions

Trapdoor functions. The notion of a *trapdoor function* forms a cornerstone of modern cryptography. Trapdoor functions were introduced in the landmark paper of Diffie and Hellman [34], which put forward the concept of public-key cryptography. The most well-known realization of this concept was given several years later by Rivest, Shamir, and Adleman [71], based on modular arithmetic. Informally, a trapdoor function is a (injective) function that is easy for anyone evaluate in the forward direction, but for which evaluation in the backward direction requires a secret input called the *trapdoor*.

One-wayness. More specfically, the classical security notion for trapdoor functions is called *one-wayness*. Intuitively, one-wayness means that given a point in the range (but not the trapdoor!), it is hard to figure out the preimage. Note that the preimage always *exists* in an absolute sense, so "hard" here refers to computational hardness, i.e., that computing it requires an infeasibly long period of time. (This distinction is akin to that between ontological and epsitemological existence in philosophy.) One immediately sees, however, that there some subtleties in how one-wayness is formalized. In particular, inversion cannot be computationally hard for *every* point in the range. Indeed, consider the algorithm that just outputs some fixed point in the domain. Then for *some* point in the range, this algorithm outputs the correct answer. Instead, one-wayness is formalized as saying that inversion is hard with high probability over the random choice of a point in the range. (One should think of this point as sampled by first choosing a random point in the domain and then applying the function.)

Application to encryption. Diffie and Hellman suggested trapdoor functions for use as public-key encryption schemes directly. That is, the encryption of a message is its image under the trapdoor function. Early on, however, Lipton [56] and Goldwasser and Micali [47] raised serious objections to such usage. Specifically, despite their power, it was noticed that trapdoor functions have certain "undesirable" properties for use as an encryption scheme. In particular, trapdoor functions such as RSA [71] *leak partial information* about their input. That is, given a random point in range, it may be hard to compute the preimage, but it may very well be feasible to recover some information about it. (In the specific case of RSA, the Jacobi symbol of the input is leaked.) Additionally, while trapdoor functions may be hard to invert over a random choice from the entire range, they may in fact be easy to invert over a random choice from a *subset* of the range, say the points whose bit representation forms ASCII character text.

1.2 Motivation for Stronger Security Notions

Goldwasser and Micali proposed addressing the weaknesses of using trapdoor functions for encryption by using *probabilistic encryption* instead. In a probabilistic encryption scheme, the encryption algorithm generates fresh random coins each time a message is encrypted. (So, encryption is now one-to-many, but decryption still recovers the unique preimage.) Indeed, the notion of *semantic security* for encryption they put forward — which roughly says that given the encryption of any message it is hard to guess any function of that message — inherently *requires* the scheme to be probabilistic. However, good randomness is not always easy to come by in practice, especially on resource constrained devices. Moreover, when securing legacy protocols, using randomness for encryption may result in an unacceptable increase to ciphertext length. It thus becomes an interesting question to what extent randomness is actually *necessary* for good encryption. In particular, could not it be possible to design a trapdoor function that hides all partial information about its input, without using randomness?

Additionally, using a trapdoor function for encryption turns out to provide some attractive properties in emerging applications. When thinking of cryptography, many might think of the classical problem of secure communication over an public channel, where Alice wants to communicate a private message to Bob, but Eve is sitting on the wire and can read any transmission she sends. With the rise of new computing environments such as cloud computing, however, the situation can be much more complicated. For instance, Bob may not store any of his data locally, but rather on a remote service that he asks only for pieces of his data as needed. Thus, it is desirable for the remote service to be able to efficiently process an encrypted search query from Bob. In the database community, this setting is known as an *outsourced database* or database-as-service [49], and how to integrate it with encryption is of increasing practical interest (see, e.g., [48, 51]). Or, encryption might be used for files in a distributed file system, such as the Tahoe file system [1]. An important property of such systems is being able to detect duplicate files efficiently, in order to avoid wasted space. (The problem of designing an encryption scheme that allows this was first introduced in [2].) Since the same plaintext is encrypted to the same ciphertext, using a trapdoor function rather than a probabilistic encryption scheme provides the needed functionalities in these applications. In the first case, the remote service is able to index Bob's encrypted files in a standard data structure, such that if Bob submits an encrypted file the server can locate it (and any associated data) in the data structure efficiently. Similarly, in the second case duplicate files can be detected efficiently by the system.

Finally, motivation for studying strengthenings to one-wayness comes from the stronger security properties desired from higher-level cryptographic primitives. Historically, one-way trapdoor functions played a central role in the construction of semantically secure encryption; in particular, it was shown that such a scheme could be constructed in a simple, black-box way from any one-way trapdoor function [15, 46]. Here "black-box" means that any trapdoor function will do, and moreover the construction does not depend on the implementation details of the latter (which typically incurs a huge efficiency cost).

However, semantic security only captures privacy against a *passive* adversary. In practical applications, adversarial parties often have considerably more power. For example, they may be able to inject their *own* packets into the network based on observed communication and see how the other parties react. This might help them learn what encrypted messages the latter are sending. A stronger notion of security capturing such *chosen-ciphertext* attacks — which additionaly allows the adversary to ask for the decryptions of some ciphertexts of its choice — was introduced by Rackoff and Simon [69], and has since become the "gold standard" for security of encryption.

Constructing encryption schemes secure against chosen-ciphertext attacks seems to be much harder than sematically secure encryption. So far, cryptographers have not found any generic construction (black-box or otherwise) based on one-way trapdoor functions. Intial constructions were instead based on more exotic objects called "non-interactive zero-knowledge proofs" [63], and then came more practical schemes from number theoretic assumptions, starting with [28]. However, Peikert and Waters [68], and subsequently Rosen and Segev [72], recently proposed novel strengthenings to the notion of one-wayness for trapdoor functions and showed that these *do* yield simple, black-box constructions of chosen-ciphertext secure encryption. Still, an underlying "theory" of these proposed strengthenings seems lacking, and it is not clear whether weaker properties of a trapdoor function would suffice for such a construction. (We stress that, while this motivates looking at weaker properties than lossiness or security under correlated products, we are still interested in notions stronger than classical one-wayness.)

1.3 Our Goals and Approach

Our goal is to study strengthenings to the classical notion of one-wayness for trapdoor functions. Specifically, we consider the following (informal) ways in which the notion of one-wayness may be strengthened:

Input entropy: The input on which the function is evaluted may be drawn from a non-uniform distribution.

Input correlations: The adversary may be given the evaluation of the function on multiple, related inputs.

Information hiding: It should be hard for the adversary to guess even "partial information" about the function's input.

Adaptivity: The adversary may have access to an inversion oracle for the function, which it may query on some points in the range.

We would like to know how these properties can be defined, whether they can be achieved, and what applications they have. Our approach to studying these questions is based on *provable security*, the foundations of which were laid by Blum, Goldwasser, Micali, and others in the 1980s (see, e.g., [15, 47]). At a high level, there are two phases of this approach, which we may call *definitional* and *constructive*.

In the definitional phase, we first provide a *syntactic* definition of the type of object we want to study, i.e., the algorithms that comprise it. This allows us to delineate the design space. We then give a *correctness* definition, i.e., the functionality that these algorithms should fulfill for their intended usage. Finally, we design a *security* definition or experiment involving an adversary against a candidate implementation of the object. In this definition, the adversary is allowed to interact with the candidate implementation in certain ways, and we define by what outcomes the adversary is said to "win." Such outcomes are meant to capture a security breach.

In the constructive phase, we design a realization of this object for which we can *prove* that it meets our definitions. However, we typically cannot prove an object meets the security condition unconditionally; even the existence of a one-way function implies P $\neq$ NP, a long-standing open problem in complexity theory. Therefore, we prove conditional security, based on the assumption that some computational problem is intractible. In other words, a security proof is a *reduction* showing that a successful adversary against the realization can be used to solve the underlying computational problem.

Classically, the treatment of security was *asymptotic*; security of a cryptographic object was formulated as no "probabilistic polynomial-time" adversary being able to win in the security experiment with "non-negligible" probability. In this thesis, we follow a more precise approach advocated by Bellare and Rogaway in the 1990s (see, e.g., [12]) where we do not define what security means formally. Rather, our security theorems are "concrete," meaning they explicitly relate the amount of resources consumed by a given adversary to the amount of resources needed to solve the underlying computational problem via our reduction. In particular, this allows practitioners to determine the values of the scheme's parameters needed to ensure security in a given appliction.

Another component of their approach we use in some cases is the random oracle model [11]. The random oracle model is a model of computation in which all parties have access to one or more oracles that implement truly random functions. One first carries out the design of schemes and security proofs in this model, and then to implement the scheme in practice one heuristically "instantiates" the oracles in certain ways using cryptographic hash functions like SHA1. Note that there is no formal evidence that the resulting scheme is secure, which is why the instantiation is a heuristic. In particular, a line of work starting with Canetti et al. [23] has identified certain (contrived) schemes that are secure in the random oracle but but trivially insecure under *any* instantiation. Therefore, the random oracle model must be used with care. When we use the random oracle model, we pay particular attention to how we are using it and what we are able to accomplish without it.

1.4 Results

We present our results along two central threads. First, we introduce a notion called *deterministic encryption*, which, intuitively, is public-key encryption that is "as-secure-as-possible" subject to the constraint that the encryption algorithm not use randomness. (Note that, syntactically, deterministic encryption is thus the same thing as a trapdoor function, but we use the term "encryption" to emphasize its intended usage.) This notion has applications we already mentioned, namely fast equality search on encrypted data and securing legacy protocols. Moreover, follow-up work to ours [6] explored the application of our notion to "hedging" probabilistic encryption schemes against bad randomness.

Second, we introduce a notion of *adaptivity* for trapdoor functions, which roughly says that the function should remain one-way even if the adversary is allowed to ask for the inverse of some points in the range. The main application is to the construction of efficient, black-box chosen ciphertext secure (probabilistic) encryption. In particular, our notion of adaptivity serves to weaken the general assumptions known to imply the latter and unify prior schemes.

1.4.1 Results on Deterministic Encryption

A SECURITY DEFINITION. We first define a security notion deterministic encryption should achieve. We provide a semantic-security style notion called DET-CPA, namely one that asks that no partial information about the plaintexts is leaked by encryption, while taking into account two inherent limitations of deterministic encryption. First, no privacy is possible if the plaintext is known to come from a small space. Indeed, knowing that c is the encryption under public key pk of a plaintext x from a set X, the adversary can compute the encryption c_x of x under pk for all $x \in X$, and return as the decryption of c the x satisfying $c_x = c$. We address this by only requiring privacy when the plaintext is drawn from a space of large min-entropy. Second, and more subtle, is that the ciphertext itself is partial information about the plaintext. We address this by only requiring non-leakage of partial information when the plaintext and partial information do not depend on the public key. This is reasonable because in real life public keys are hidden in our software and data does not depend on them. Our definition also explicitly models privacy for multiple related messages, requiring that each individual message has large min-entropy. Additionally, it extends to chosen-ciphertext security by giving the adversary a decryption oracle; however, we just address chosen-plaintext security for now, leaving chosen-ciphertext security for the part of the thesis about adaptive trapdoor functions.

A GENERIC CONSTRUCTION IN THE RO MODEL. We next turn to constructions meeting our new notion. Our first construction is generic and natural: Deterministically encrypt plaintext x by applying the encryption algorithm of a randomized scheme but using as coins a hash of (the public key and) x. We show that this "Encrypt-with-Hash" deterministic encryption scheme is DET-CPA secure in the random oracle (RO) model of [11] assuming the starting randomized scheme is IND-CPA secure. We note that a similar idea was used by [8] to show that IND-CPA secure randomized encryption scheme implies *one-way* trapdoor functions in the RO model; our results show that via this technique we can actually achieve a much stronger security notion.

A LENGTH-PRESERVING CONSTRUCTION IN THE RO MODEL. Our second construction is an extension of RSA-OAEP [12] that we call RSA-DOAEP ("D" for deterministic). Unlike OAEP, the padding transform is deterministic, using part of the message in place of the randomness in OAEP, and uses three Feistel rounds rather than the two of OAEP. RSA-DOAEP is proven DET-CPA secure in the RO model assuming RSA is one-way. This construction has the attractive feature of being length-preserving. (The length of the ciphertext equals the length of the plaintext.) This is important when bandwidth is expensive —senders could be power-constrained devices— and for securing legacy code.

AN INDISTINGUISHABILITY-BASED NOTION. Towards constructing deterministic encryption schemes in the standard model, we next revisit our DET-CPA security notion. The DET-CPA security notion captures our intuition about security of determinsitic encryption well but is difficult to work with. Moreover, having alternative but equivalent definitions of security increases our confidence that we have the "right" one. Following Goldwasser and Micali [47], who in addition to semantic security formulated an indistinguishability-based notion for probabilistic encryption that was later shown equivalent [59], we would like to find a simpler indistinguishability-based definition of security for deterministic encryption. Fortunately, Dodis and Smith [38] already formulated such a definiton in the context of information-threoretically secure, one-time symmetric encryption for high-entropy messages, which we can adapt straightforwardly to our context. The definition, which we call DH-CPA (for "distribution hiding"), asks that it be hard to distinguish the encryptions of two messages drawn from any two high min-entropy distrbutions on the plaintext space, respectively. We prove that DET-CPA security is equivalent to DH-CPA security for plaintexts of slightly lower min-entropy. Note that this entropy loss is in some sense inherent, since the definitions certainly cannot be equivalent for uniformly random messages (in this case, every scheme is trivially DH-CPA secure).

Single or multiple messages. We also examine relations between single and multiple-message security for deterministic encryption. In the classical setting of probabilistic encryption, a hybrid argument proves equivalence of secuity in these cases [4]. This hybrid argument fails in the case of deterministic encryption, however, and in fact we show the two definitions are not equivalent. Thus, in general we must explicitly model the encryption of multiple messages, and it also becomes interesting to consider weaker variants of the definition that relax the requirements on the message distributions. In particular, Fehr [41] has shown that the equivalence between security for single and multiple messages is recovered when multi-message security is considered for *block-sources*, that is, where each message has sufficient "fresh" entropy conditioned on the others. We also consider q-bounded deterministic encryption, where at most q arbitrarily correlated messages are encrypted. (Our above-mentioned counter-example shows single-message and 2-bounded deterministic encryption are inequivalent; we conjecture that q-bounded and $(q+1)$-bounded deterministic encryption are also inequivalent.)

A construction from "robust" hardcore functions. The DH-CPA notion gives us a handle on how to construct deterministic encryption schemes in the standard model. To this end, we introduce a new notion we call a “robust” hardcore function. Recall that a hardcore function [15] of a trapdoor function is a function of the input whose output is indistinguishable from random, even given the image of the input under the trapdoor function. By “robust” we means that that hardcore function remains hardcore even if the entropy of the input is slightly reduced. We use this to give a generic construction of DET-CPA secure deterministic encryption from certain trapdoor functions. The scheme is similar to the Encrypt-with-Hash RO model scheme. Specifically, one encrypts plaintext x by encrypting under a randomized scheme the image of x under the trapdoor function, using the hardcore function applied to x as the coins. Using our above-mentioned equivalence between DH-CPA and DET-CPA, this “Encrypt-with-Hardcore” scheme is shown DET-CPA for any distribution on the input for which the trapdoor function admits a robust hardcore function of appropriate output length.

Instantiations. We show that in the case of a single input (or, equivalently, block-sources [41]), a suitable hardcore function can be obtained from exponentially-hard one-way trapdoor permutations or trapdoor functions, via the Goldreich-Levin function [46]. (Trapdoor permutations require less hardness; in particular, in the case of standard one-way trapdoor permutations we obtain a construction for nearly uniform inputs.) We also consider *lossy* trapdoor functions, introduced by Peikert and Waters [68]. Intuitively, lossy trapdoor functions have a description that is indistinguishable from that of a function that loses information about its input (i.e., has a bounded range). Peikert and Waters showed that a universal hash function is hardcore for a lossy trapdoor function, and we show that it additionally has the properties we need here. We thereby obtain instantiations of the Encrypt-with-Harcore scheme under these assumptions secure for encrypting a single high-entropy message (or block-sources [41]).

Improved constructions from lossy trapdoor functions. In fact, based on lossy trapdoor functions we give improved constructions, both in terms of efficiency and the plaintext distributions for which they achieve security. The improvements are based on a paradigm introduced by Dodis and Smith [38], who showed that in addition to producing a uniform output randomness extractors actually hide all partial information about their input. Indeed, this follows in our context from the DH-CPA and DET-CPA equivalence. To apply this idea to deterministic encryption, we use lossy trapdoor functions where the lossy mode has some randomness extraction property. For example, if the lossy mode is a universal hash, then security under our DH-CPA notion follows by the Leftover Hash Lemma [50]. (The latter says that any small-range universal hash "smoothes out" any high-entropy source to nearly uniform on the range.) To use standard lossy trapdoor functions, we can first pre-process the input plaintext with a pairwise independent permutation and appeal to the "Crooked" Leftover Hash Lemma [37] for security, which says that a pairwise-independent permutation composed with any small range function f looks like f applied to uniform input. By strengthening to the Crooked Leftover Hash Lemma, we show that to acheive q-bounded security (i.e., security for up to q arbitrarily correlated high min-entropy plaintexts) we can use a $2q$-wise independent permutation instead (More generally, the resulting construction achieves security for q-block-sources, where every q messages "fresh" entropy is introduced.) However, permutations with independence greater than 3-wise are not known to exist. Fortunately, Kaplan et al. [52] give good constructions of *almost* q-wise independent permutations that we show suffices for our purposes.

Our high-level results on determinsitic encryption are summarized in Figure 1. In the figure, dashed implications are trivial. Regarding assumptions, "IND-CPA+RO" means IND-CPA secure encryption and in the RO model, "LTDF" means lossy trapdoor functions, and "EXH-TDF" means exponentially-hard (in terms of one-wayness)

$$\begin{tikzcd}
\text{IND-CPA + RO} \arrow[d] & \text{LTDF} \arrow[d] & \text{EXH-TDF} \arrow[d] \\
\text{DH-CPA} \arrow[r, dashed] \arrow[d, shift left] & \text{DH-CPA-}q \arrow[r, dashed] \arrow[d, shift left] & \text{DH-CPA-1} \arrow[ll, bend right, "/" marking] \arrow[d, shift left, "{[41]}"] \\
\text{DET-CPA} \arrow[u, shift left] & \text{DET-CPA-}q \arrow[u, shift left] & \text{DET-CPA-BS} \arrow[u, shift left]
\end{tikzcd}$$

Figure 1: Relations between cryptographic assumptions and notions of security for deterministic encryption.

---

trapdoor function. For simplicity the figure does not consider chosen-ciphertext attacks.

1.4.2 Results on Adaptive Trapdoor Functions

Adaptivity for trapdoor functions. Our second strengthening to one-wayness for trapdoor functions is a notion we call *adaptive* trapdoor functions. Loosely speaking, adaptive trapdoor functions remain one-way even when the adversary is given access to an inversion oracle, which it may query on points other than its challenge (the point it needs to invert). We also introduce a natural extension we call tag-based adaptive trapdoor functions, which in addition to the normal input also take a tag. For tag-based adaptive trapdoor functions, the adversary may query its oracle on any point, but on a tag other than the challenge one.

Chosen-ciphertext security. We next give a black-box construction of chosen-ciphertext secure encryption from adaptive trapdoor functions and from adaptive tag-based trapdoor functions. While constructing chosen-ciphertext secure public-key encryption from tag-based adaptive trapdoor functions is straightforward, constructing the former from adaptive trapdoor functions turns out to be more subtle. Namely, we first obtain a such a scheme for a single-bit message using the hardcore bit of the trapdoor function, but unlike for the classical constructions of semantically-secure encryption based on one-way trapdoor functions [15] it is important here that the ciphertext not contain the message xor'ed with the latter; rather the message is encrypted *as* the hardcore bit itself. By a recent result of Myers and Shelat [61], this construction implies a black-box many-bit scheme as well. On the other hand, hybrid encryption (where the public-key encryption scheme is only used to encrypt a symmetric key, the latter being used to encrypt the actual message) permits a much more efficient direct construction of such a scheme in the case that the adaptive trapdoor function is a permutation or has linearly many simultaneous hardcore bits.

Adaptive trapdoor functions in the RO model. In the random oracle model [11], the notions of adaptive one-wayness and one-wayness are equivalent in the sense that each such function can be constructed from the other. In fact, we show that the Encrypt-with-Hash determinstic encryptions scheme is DET-CCA, which is a strengthening to the notion of adaptive one-wayness, if its starting randomized scheme is only IND-CPA but meets a minor extra condition, namely that no ciphertext occurs with too high a probability over the choice of the randomness for encryption. This turns out to be a quite mild condition satisfied by all practical schemes.

Adaptive trapdoor functions in the standard model. To construct adaptive trapdoor functions in the standard model, we examine their relation to the notions of correlated-product [72] and lossy trapdoor functions [68]. Intuitively, correlated-product trapdoor functions remain one-way even if the adversary sees many independent instances of the function evaluated on the same random input. Inspired by the constructions of chosen-ciphertext secure public-key encryption in [68, 72] (which are based on earlier work by Dolev et al. [39]), we show simple, black-box constructions of both adaptive trapdoor functions and adaptive tag-based trapdoor functions from correlated-product trapdoor functions. Since as shown in [72, 60], lossy trapdoor functions imply the latter, this also gives us adaptive trapdoor functions from lossiness. However, we go on to show that adaptivity is sufficiently general to allow a much more efficient direct constructions using an all-but-one trapdoor functions, an extension to lossy trapdoor functions introduced in [68] as well. In fact, we show this construction achieves a stronger notion of "adaptive lossiness," which, when plugged into our constructions of deterministic encryption based of lossy trapdoor functions, serves to "upgrade" these schemes from chosen-plaintext to chosen-ciphertext security without random oracles.

Unifying prior work. Additionally, applying our generic construction of chosen-ciphertext secure encryption from adaptive tag-based trapdoor functions to the above constructions yields precisely chosen-ciphertext secure shemes of [72] and [68], respectively. This means that these works were implicitly constructing adaptive tag-based trapdoor functions, and that the latter notion "abstracts out" a particular aspect of their constructions not formalized before. This unifies and clarifies their schemes from a conceptual standpoint. It also leads to optimized schemes based on a transformation proposed by Boneh et al. [17]. A noteworthy feature of the optimized schemes is that they are fully "witness-recovering" as defined in [68]; that is, via the decryption process the receiver is able to recover *all* of the randomness used by the sender for encryption. (The original constructs of [68, 68, 72] technically do not achieve this since, as the authors note, the receiver does not recover the coins used to generate one-time signature keys in their scheme.)

A black-box separation from correlated-products. Recently, Vahlis [76] showed that there is no black-box construction of correlated-product trapdoor functions from one-way trapdoor functions. We next observe that his result extends to rule out a black-box construction of the former from adaptive trapdoor functions as well, by using the same "breaking" oracle. This does not immediately rule out a black-box construction of correlated-product trapdoor functions from tag-based adaptive trapdoor functions, but we also rule this out by giving a construction of tag-based adaptive trapdoor functions from exponentially-hard adaptive trapdoor functions (see below); the latter is separated from correlated-product trapdoor functions by our extension of Vahlis's result as well. Combined with the above-mentioned constructions, this means that, surprisingly, adaptive trapdoor functions and tag-based adaptive trapdoor functions are *strictly weaker* than correlated-product trapdoor fuctions and lossy trapdoor functions, cf. Figure 2. In the figure, $\rightarrow$ is an implication while $\not\rightarrow$ is a black-box separation. Dashed lines indicate trivial implications.s The considered notions for TDFs are extractable TDF (EX-TDF), lossy TDF, correlated-product TDF (CP-TDF), one-more TDF (OM-TDF), and one-way TDF (OW-TDF). Whether adaptive trapdoor functions can be shown weaker than standard (one-way) trapdoor functions remains an interesting open question.

Relations between tag-based and non-tag-based adaptivity. To better understand the relation between adaptive trapdoor function and adpative tag-based trapdoor functions, we note that tag-based trapdoor functions can be viewed as trapdoor functions where part of the input is output in the clear. Using this observation, we show that adaptive trapdoor functions and adaptive tag-based trapdoor functions are equivalent under *exponential hardness*, meaning an exponentially-hard version of one primitive implies an exponentially hard version of the other. Whether the equivalence holds in general remains open.

Adaptivity from RSA. Finally, we show that tag-based adaptive trapdoor functions are realizable from specific assumptions not known to imply correlated-product trapdoor functions. Namely, we consider the "instance-independent" RSA assumption (II-RSA) introduced (in a more general form) by Paillier and Villar [66]. Roughly, the assumption is that solving an RSA challenge $y = x^e \bmod N$ remains hard even when the adversary is given access to an inversion oracle that on input (y', e') returns

$$\begin{tikzcd}
& \text{EX-TDF} \arrow[dr, dashed] & & & \\
\text{Lossy TDF} \arrow[r, shift left] & \text{CP-TDF} \arrow[l, shift left, "/" marking, "{[72]}"'] \arrow[r, shift left] & \boxed{\text{ATDF}} \arrow[l, shift left, "/" marking] \arrow[r, dashed] \arrow[d] & \text{OM-TDF} \arrow[r, dashed] & \text{OW-TDF} \arrow[lll, bend right, "/" marking, "{[76]}"'] \\
& & \text{CCA PKE} & &
\end{tikzcd}$$

Figure 2: Relations between the various security notions for trapdoor functions and CCA-secure PKE, centered around adaptive trapdoor functions.

$y'^{1/e'} \bmod N$, where $e \neq e'$ are primes. We show that II-RSA gives rise to a tag-based adaptive trapdoor functions. This also leads to a very efficient chosen-ciphertext secure RSA-based public-key encryption scheme in the standard model.

1.5 Related and Follow-Up Work

Prior to our intial work on deterministic encryption, we are not aware of any general study of stronger notions of security for trapdoor functions.[1] However, an earlier line of research looked at properties stronger than one-wayness that seem to be satisfied by plain RSA. In particular, Bellare et al. [10] defined the "one-more RSA" problem in order to argue security of Chaum's blind signature scheme. The "one-more" property can actually be considered a general security notion for trapdoor functions, although the authors did not treat it at this level of generality. Additionally, our work treats properties of trapdoor functions that are not satisfied by "conventional" schemes like plain RSA.

In the symmetric setting, deterministic encryption is both easier to define and to achieve than in the asymmetric setting. Consider the experiment that picks a random challenge bit b and key K and provides the adversary with a left-or-right oracle that, given plaintexts x_0, x_1 returns the encryption of x_b under K. Security asks that the

[1]The works of [68, 72] came afterwards.

adversary find it hard to guess b as long as its queries $(x_0^1, x_1^1), \ldots, (x_0^q, x_1^q)$ satisfy the condition that $x_0^1, \ldots, x_0^q$ are all distinct and also $x_1^1, \ldots, x_1^q$ are all distinct. To the best of our knowledge, this definition of privacy for deterministic symmetric encryption first appeared in [9]. However, it is met by a pseudorandom permutation and in this sense deterministic symmetric encryption goes back to [57].

Our work on deterministic encryption builds on that on perfectly one-way probabilistic hash functions (POWHFs) [21, 25] and information-theoretically secure one-time symmetric encryption (also called "entropic security") [73, 38]. These works also consider security notions that hold for high min-entropy inputs. The first however cannot be met by deterministic schemes, and neither handle the public-key related subtleties of deterministic public-key encryption (namely that we cannot protect functions of the message depending on the public key) or correlated messages.

Regarding the application to efficient search on encrypted data, we note that Boneh et al. [18] introduced an earlier notion of "public-key encryption with keyword search" designed for applcations such as secure email routing. When applied to the database setting it results in linear time (in the size of the database) search, whereas a large body of work in the database community indicates they want more efficient solutions (see e.g. [48, 51]), which deterministic encryption addresses.

Our work on adaptive trapdoor functions builds on the work of Pandey et al. [67], who introduced a notion they called "adaptive one-way functions," although their notion would be more accurately referred to as adaptive *tag-based* one-way functions. Besides the obvious difference of not having a trapdoor (the inversion oracle in their security experiment is unbounded), their notion differs from ours in that *it does not have a public key.* This is crucial for the applications of [67] to non-malleable commitment but also makes it much harder to construct. Indeed, they are not known to be realizable based on any standard assumptions.

Additionally, in [22] Canetti and Dakdouk define the notion of extractable trapdoor functions, which essentially says that no efficient adversary can compute $f(x)$ without "knowing" x (similar to the notion of plaintext-awareness for probabilistic encryption [12]). This notion implies adaptivity but unfortunately no instantiation based on standard assumptions is known (the authors only provide constructions of extractable one-way functions, without a trapdoor).

Following our work on deterministic encryption, Bellare et al. [6] explored the applications of our notion to designing "hedged" public-key encryption, a proabilistic encryption scheme that "falls back" to determinsitic-encryption style security when randomness is buggy. This was motivated in part by a bug in the Debian Linux version of OpenSSL that caused the system to generate predictable randomness [79]. Additionally, Dent et al. [32] applied our notions to the design of "confidential" signature schemes, which can be combined in an encrypt-and-sign way with digital signature schemes such that the message remains hidden.

Following our work on adaptive trapdoor functions, Wee [77] introduced a notion of adaptive trapdoor *relations*, which are like adaptive trapdoor functions except the function itself is not efficiently computable but rather admits efficient sampling of a random domain point together with its image. Wee shows this notion also implies black-box chosen-ciphertext secure public-key encryption and furthermore admits instantiations from standard computational assumptions like RSA and computational Diffie-Hellman.

1.6 Organization and Credits

The remainder of this thesis is organized as followed. Chapter 2 introduces the notation and conventions we will use throughout, as well as basic cryptographic and statistical notions we will need. Chapter 3 introduces determinisitic encryption, providing a security definition and constructions in the random oracle model. Chapter 4 studies more foundational issues for deterministic encryption, providing definitoinal equivalences and constructions without random oracles. Chapter 5 studies adaptive trapdoor functions and their applications to chosen-ciphertext security.

The material in Chapter 3 is based on joint work with Mihir Bellare and Alexandra Boldyreva [5]. The material in Chapter 4 is based on joint work with Alexandra Boldyreva and Serge Fehr [16] (which itself builds on an earlier work of Fehr [41]) and with Mihir Bellare, Marc Fischlin, and Thomas Ristenpart [7]. It additionally draws on material from a manuscript of ours [64] that generalizes and strengthens these results and is currently unpublished. The material in Chapter 5 is based on joint work with Eike Kiltz and Payman Mohassel [54].

We take full responsibility for any errors appearing in this thesis. The reader is encouraged to consult the public versions of any relevant publications for possible clarifications or corrections.

CHAPTER II

PRELIMINARIES

In this chapter, we collect the notation and conventions we will use throughout the thesis as well as useful statistical and cryptographic notions.

2.1 Notation and Conventions

An adversary is either an algorithm or a tuple of algorithms. Unless otherwise indicated, an adversary or algorithm may be randomized. By convention, the running-time of an adversary includes both its actual running-time and the time to run its overlying experiment. We denote by $\mathbb{N}$ the set of natural numbers and by $\mathbb{Z}$ the set of integers. For $n \in \mathbb{N}$, we denote by $\mathbb{Z}_n$ the ring of integers modulo n and by $[n]$ the set $\{1, \ldots, n\}$. The security parameter is denote by k, and 1^k denotes the string of k ones. We sometimes surpress dependence of variables on k for readability.

Algorithmic notation. If A is an algorithm then $x \overset{\$}{\leftarrow} A(\ldots)$ denotes that x is assigned the output of running A on the elided inputs and a fresh random tape, while if S is a finite set then $s \overset{\$}{\leftarrow} S$ denotes that s is assigned a uniformly random element of S. We often use the abbreviation $x_1, \ldots, x_n \overset{\$}{\leftarrow} A(\ldots)$ for $x_1 \overset{\$}{\leftarrow} A(\ldots)\,;\, \ldots\,;\, x_n \overset{\$}{\leftarrow} A(\ldots)$, for any $n \in \mathbb{N}$, and similarly for sets. If A is deterministic then we drop the dollar sign above the arrow. We let "$A(\ldots) \Rightarrow y$" denote the event that A outputs y in the above experiment.

Strings. We denote by $\{0,1\}^*$ the set of all (binary) strings, and by $\{0,1\}^n$ the set of strings of length n, for any $n \in \mathbb{N}$. If x is a string then $|x|$ denotes its length in bits and $x[i \ldots j]$ denotes bits i through j of x, for any $1 \leq i \leq j \leq |x|$. By $x_1 \| \cdots \| x_n$ we denote an encoding of strings $x_1, \ldots, x_n$ from which $x_1, \ldots, x_n$ are uniquely recoverable. We denote by $x \oplus y$ the bitwise exclusive-or (xor) of equal-length strings x, y. An n-bit string may also be interpreted as an n-dimensional vector of $GF(2)$. In particular, for two n-bit strings x, y we denote by $\langle x, y \rangle$ the inner-product of x and y over $GF(2)$.

Vectors. Vectors are denoted in boldface, for example $\mathbf{x}$. If $\mathbf{x}$ is a vector then $|\mathbf{x}|$ denotes the number of components of $\mathbf{x}$ and $\mathbf{x}[i]$ denotes its ith component, for any $1 \le i \le |\mathbf{x}|$. For convenience, we extend algorithmic and functional notation to operate on a vector of inputs component-wise. That is, if A is an algorithm and $\mathbf{x}, \mathbf{y}$ are vectors then $\mathbf{z} \overset{\$}{\leftarrow} A(\ldots, \mathbf{x}, \ldots, \mathbf{y}, \ldots)$ denotes that $\mathbf{z}[i] \overset{\$}{\leftarrow} A(\ldots, \mathbf{x}[i], \ldots, \mathbf{y}[i], \ldots)$ for all $1 \le i \le |\mathbf{x}|$, where the elided inputs are fixed across all invocations.

2.2 Statistical Notions

We write P_X for the distribution of random variable X and $P_X(x)$ for the probability that X puts on value x, i.e., $P_X(x) = \mathrm{P}[X = x]$. The *min-entropy* of X is $\mathrm{H}_\infty(X) = -\log(\max_x P_X(x))$. The *conditional min-entropy* of X given Y is defined as $\mathrm{H}_\infty(X|Y) = -\log(\max_{x,y} P_{X|Y=y}(x))$. The *average conditional* min-entropy of X given Y [36] is defined as $\tilde{\mathrm{H}}_\infty(X|Y) = -\log(\sum_y P_Y(y) \max_x P_{X|Y=y}(x))$. The *statistical distance* between random variables X and Y is given by $\Delta(X, Y) = \frac{1}{2} \sum_x |P_X(x) - P_Y(x)|$. If $\Delta(X, Y)$ is at most ε then we say X, Y are *ε-close* and write $X \approx_\varepsilon Y$. The *collision probability* of random variable X is $\sum_x P_X(x)^2$, and the *square of the 2-distance* is $D(X, Y) = \sum_x \big(P_X(x) - P_Y(y)\big)^2$. Note that for any random variable X, $\mathrm{Col}(X) \le \mathrm{H}_\infty(X)$. We also use that $\Delta(X, Y) \le \frac{1}{2}\sqrt{|\mathcal{X}| \cdot D(X, Y)}$, which follows immediately from the Cauchy-Schwarz inequality.

t-wise independent functions. Let $F \colon \mathcal{K} \times D \to R$ be a function. We say that F is *t-wise independent* if for all distinct $x_1, \ldots, x_t \in D$ and all $y_1, \ldots, y_t \in R$

$$\Pr\Big[\, F(K, x_1) = y_1 \ \wedge\ \ldots\ \wedge\ F(K, x_t) = y_t \ :\ K \overset{\$}{\leftarrow} \mathcal{K} \,\Big] \;=\; \frac{1}{|R|^t}\,.$$

In other words, $F(K, x_1), \ldots, F(K, x_t)$ are all uniformly and independently random over R. 2-wise independence is also called *pairwise independence*. We say that F is *universal* if for all distinct $x_1, x_2 \in D$

$$\Pr\left[\, F(K, x_1) = F(K, x_2) \;:\; K \xleftarrow{\$} \mathcal{K} \,\right] \;=\; \frac{1}{|R|} \,.$$

Note that universality is a weaker property than pairwise independence.

Leftover hash lemma. The classical Leftover Hash Lemma of Håstad et al. [50] says that a compressing univeral hash function "smoothes out" a high-entropy source to essentially uniform.

Lemma 2.2.1. (Leftover Hash Lemma) [50] Let $H \colon \mathcal{K} \times D \to R$ be a universal function. Let X be a random variable over D. Then

$$\Delta\big((K, H(K, X)), (K, U)\big) \le \frac{1}{2}\sqrt{|R| \cdot \mathrm{Col}(X)} \,,$$

where $K \xleftarrow{\$} \mathcal{K}$ and U is uniform and independent on R.

2.3 Cryptographic Primitives

Public-key encryption. A *public-key encryption scheme* with plaintext-space PtSp is a triple of algorithms $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. The key-generation algorithm $\mathcal{K}$ takes input 1^k to return a public key pk and matching secret key sk. We assume for convenience that sk contains pk. The encryption algorithm $\mathcal{E}$ takes pk and a plaintext m to return a ciphertext. The deterministic decryption algorithm $\mathcal{D}$ takes sk and a ciphertext c to return a plaintext. We require that for all plaintexts $m \in \mathrm{PtSp}$

$$\Pr\left[\, \mathcal{D}(sk, \mathcal{E}(pk, m)) = m \;:\; (pk, sk) \xleftarrow{\$} \mathcal{K}(1^k) \,\right] \;=\; 1 \,.$$

In this thesis, we deal almost exclusively with public-key encryption. Therefore, when we say "encryption scheme" we mean "public-key encryption scheme." Additionally, we use the terms "message" and "plaintext" interchangeably.

Chosen-plaintext security. Next we define security against chosen-plaintext attack for encryption [47], which captures the intuition that no information about the message is leaked from the ciphertext. To an encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ and an adversary $A = (A_1, A_2)$ we associate

```
**Experiment** $\mathbf{Exp}^{\text{ind-cpa}}_{\Pi,A}(k)$:
$b \xleftarrow{\$} \{0,1\}$ ; $(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
$(m_0, m_1, state) \xleftarrow{\$} A_1(pk)$
$c \xleftarrow{\$} \mathcal{E}(pk, m_b)$
$d \xleftarrow{\$} A_2(pk, c, state)$
If $d = b$ return 1 else return 0
```

where we require A_1's output to satisfy $|m_0| = |m_1|$. Define the *IND-CPA advantage* of A against Π as

$$\mathbf{Adv}^{\text{ind-cpa}}_{\Pi,A}(k) = 2 \cdot \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa}}_{\Pi,A}(k) \Rightarrow 1\,\right] - 1 \ .$$

Equivalently,

$$\mathbf{Adv}^{\text{ing-cpa}}_{\Pi,A}(k) = \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa}}_{\Pi,A}(k) \Rightarrow 1 \mid b = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa}}_{\Pi,A}(k) \Rightarrow 1 \mid b = 0\,\right] \ .$$

Chosen-ciphertext security. Security against chosen-ciphertext attack [69] additionally allows the adversary to submit decryption queries, capturing active attacks. Let $A = (A_1, A_2)$ be an adversary such that A_2 has access to an oracle. To an encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ and A we associate

```
**Experiment** $\mathbf{Exp}^{\text{ind-cca}}_{\Pi,A}(k)$:
$b \xleftarrow{\$} \{0,1\}$ ; $(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
$(m_0, m_1, state) \xleftarrow{\$} A_1(pk)$
$c \xleftarrow{\$} \mathcal{E}(pk, m_b)$
$d \xleftarrow{\$} A_2^{\mathcal{D}(sk,\cdot)}(pk, c, state)$
If $d = b$ return 1 else return 0
```

where we require A_1's output to satisfy $|m_0| = |m_1|$ and that A_2 does not query c to its oracle. Define the *IND-CCA advantage* of A against Π as

$$\mathbf{Adv}^{\text{ind-cca}}_{\Pi,A}(k) = 2 \cdot \Pr\left[\, \mathbf{Exp}^{\text{ind-cca}}_{\Pi,A}(k) \Rightarrow 1 \,\right] - 1 \; .$$

Equivalently,

$$\mathbf{Adv}^{\text{ing-cca}}_{\Pi,A}(k) = \Pr\left[\, \mathbf{Exp}^{\text{ind-cca}}_{\Pi,A}(k) \Rightarrow 1 \;\mid\; b = 1 \,\right] - \Pr\left[\, \mathbf{Exp}^{\text{ind-cca}}_{\Pi,A}(k) \Rightarrow 1 \;\mid\; b = 0 \,\right] \; .$$

It will sometimes be convenient for us to consider IND-CPA and IND-CCA adversaries $A = (A_1, A_2)$ that output vectors, meaning A_1 outputs $(\mathbf{x}_0, \mathbf{x}_1, \mathit{state})$ and A_2 takes input $(\mathcal{E}(pk, \mathbf{x}_b), \mathit{state})$. (Recall that $\mathcal{E}(pk, \mathbf{x}_b)$ is defined component-wise.) A hybrid argument [4] implies security against such adversaries under the respective notions are equivalent to security against one-message adversaries (In fact, a stronger equivalence holds, where messages are chosen adaptively based on previously seen ciphertexts.)

Trapdoor functions and one-wayness. A *trapdoor function generator* is an algorithm $\mathcal{F}$ that on input 1^k outputs (f, f^{-1}) where f is (the description of) an injective function on $\{0,1\}^k$ and f^{-1} is (a description of) its inverse. To a trapdoor function generator $\mathcal{F}$ and inverter I we associate

```
**Experiment** $\mathbf{Exp}^{\text{owf}}_{\mathcal{F},I}(k)$:
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}$
$x \xleftarrow{\$} \{0,1\}^k$
$x' \xleftarrow{\$} I(f, f(x))$
If $x = x'$ return 1 else return 0
```

Define the *OWF advantage* of I against F as

$$\mathbf{Adv}^{\text{owf}}_{\mathcal{F},I}(k) = \Pr\left[\, \mathbf{Exp}^{\text{owf}}_{\mathcal{F},A}(k) \Rightarrow 1 \,\right] \; .$$

Hardcore bits and functions. Let hc be a function that takes as input a description of a function f and a point $x \in \{0,1\}^k$ To a trapdoor function generator $\mathcal{F}$, function hc, and distinguisher D we associate

```
Experiment $\mathbf{Exp}^{\mathrm{hcf}}_{\mathcal{F},\mathsf{hc},D}(k)$:
$b \xleftarrow{\$} \{0,1\}$ ; $(f, f^{-1}) \xleftarrow{\$} \mathcal{F}$
$h_0 \leftarrow \mathsf{hc}(f, x)$ ; $h_1 \xleftarrow{\$} \{0,1\}^n$
$d \xleftarrow{\$} D(f, f(x), h_b, r)$
If $d = b$ return 1 else return 0
```

For all $k \in \mathbb{N}$, define the *HCF advantage* of A against F, hc as

$$\mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F},\mathsf{hc},A}(k) = 2 \cdot \Pr\left[\, \mathbf{Exp}^{\mathrm{hcf}}_{\mathcal{F},\mathsf{hc},A}(k) \Rightarrow 1 \,\right] - 1 \; .$$

Note that above we allow that hardcore function to depend on the description of the TDF, which simplifies our exposition somewhat.

Goldreich-Levin function. The celebrated result of Goldreich and Levin [46] provides a hardcore function for any one-way function (by first modifying the description of the function to be of a specific "padded" form). The output length of the hardcore function depends on the computational hardness of inverting the one-way function (i.e., the more resources required to invert it, the greater the length of its hardcore function).

Namely, let $\mathcal{F}$ be a trapdoor function generator and let $\mathcal{H}\colon \mathcal{K} \times \{0,1\}^k \to \{0,1\}^n$ be a function. Define its *H-padded version* $\mathcal{F}[H]$ that on input 1^k returns $(f, K), (f^{-1}, K)$ where $(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$ and $K \xleftarrow{\$} \mathcal{K}$; evaluation is defined for $x \in \{0,1\}^k$ as $f(x)$ (i.e., evaluation just ignores K) and inversion is defined analogously. Define the *length-i Goldreich-Levin function* $\mathcal{GL}^i\colon \{0,1\}^{i \times k} \times \{0,1\}^k \to \{0,1\}^i$ as

$$\mathcal{GL}^i(M, x) = Mx$$

where Mx is the matrix-vector product of M and x over $\mathbb{Z}_2$.

Theorem 2.3.1. (**Goldreich-Levin Theorem** [46]) Let $\mathcal{F}[\mathcal{GL}^i]$ be as defined above. Let D be a distinguisher against $\mathcal{GL}^i$. Then there is a inverter I such that for all

$k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F}[\mathcal{GL}^i],\mathcal{GL}^i,D}(k) \;\leq\; 2^{i+3} \cdot \mathbf{Adv}^{\mathrm{owf}}_{\mathcal{F},I}(k)\ . \tag{1}$$

Furthermore, the running-time of I is the time for $O(\varepsilon^{-4}k^3)$ executions of D where $\varepsilon = \mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F}[\mathcal{GL}^i],\mathcal{GL}^i,D}(k)$.

Lossy trapdoor functions. A *lossy trapdoor function (LTDF) generator* [68] is a pair $\mathsf{LTDF} = (\mathcal{F}, \mathcal{F}')$ of algorithms. Algorithm $\mathcal{F}$ is a trapdoor function generator, and algorithm $\mathcal{F}'$ outputs a (description of a) function f' on $\{0,1\}^k$. We call $\mathcal{F}$ the "injective mode" and $\mathcal{F}'$ the "lossy mode" of LTDF respectively, and we call $\mathcal{F}$ "lossy" if it is the first component of some lossy TDF. For a distinguisher D, define its *LTDF advantage* against LTDF as

$$\mathbf{Adv}^{\mathrm{ltdf}}_{\mathsf{LTDF},D}(k) = \Pr\left[\, D(f) \Rightarrow 1 \;:\; (f, f^{-1}) \xleftarrow{\$} \mathcal{F} \,\right] - \Pr\left[\, D(f') \Rightarrow 1 \;:\; f' \xleftarrow{\$} \mathcal{F}' \,\right]\ .$$

We say LTDF has *residual leakage* s if for all f' output by $\mathcal{F}'$ we have $|R(f')| \leq 2^s$. The *lossiness* of LTDF is $\ell = k - s$.

RSA. An *RSA trapdoor-permutation generator* [71] is an algorithm $\mathcal{F}$ that on input 1^k returns $(N,e),(N,d)$ where N is the product of two distinct $k/2$-bit primes and $ed \equiv 1 \bmod \phi(N)$. (Here $\phi(\cdot)$ is Euler's phi function, so $\phi(N) = (p-1)(q-1)$.)

Partial one-wayness of RSA. The RSA trapdoor permutation is widely believed to be one-way. We also use a result of Fujisaki et al. [42] that one-wayness of RSA is equivalent to what they call "partial one-wayness." Partial one-wayness means that it is hard to compute the last k_0 bits of x for some k_0. Namely, to a trapdoor function generator $\mathcal{F}$ and inverter I we associate

```
Experiment $\mathbf{Exp}^{\mathrm{powf}}_{\mathcal{F},I}(k)$:
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}$
$x \xleftarrow{\$} \{0,1\}^k$
$x' \xleftarrow{\$} I(f, f(x))$
If $x[n-k_0+1 \ldots k] = x'$ return 1 else return 0
```

Define the *POWF advantage* of I against F as

$$\mathbf{Adv}^{\mathrm{powf}}_{\mathcal{F},I}(k) = \Pr\left[\,\mathbf{Exp}^{\mathrm{powf}}_{\mathcal{F},A}(k) \Rightarrow 1\,\right] .$$

Intuitively, the following lemma from [42] says that RSA is one-way if it is partial one-way and k_0 is large enough. (What [42] shows is actually more general, but we only need a specific case.)

Lemma 2.3.2. (Partial One-Wayness of RSA) [42] Let $\mathcal{RSA}$ be an RSA trapdoor permutation generator with modulus length k_1 and let I' be a partial one-way adversary against $\mathcal{RSA}$. Then there exists an inverter I against $\mathcal{RSA}$ such that

$$\mathbf{Adv}^{\mathrm{powf}}_{\mathcal{RSA},I'}(k) \;\le\; \sqrt{\mathbf{Adv}^{\mathrm{owf}}_{\mathcal{RSA},I}(k) + 2^{2k-4k_0+10}} \;+\; 2^{k-2k_0+5} .$$

Furthermore, the running-time of I is at most twice that of I' plus $O(k^3)$.

2.4 Code-Based Game Playing

Our security analyses often employ the code-based game playing technique of [14]. We recall some game-related language and conventions from [14] that we will use.

A game consists of an Initialize procedure, procedures that respond to an adversary's oracle queries, and a Finalize procedure. For an example, see Figure 4. First, the Initialize procedure executes, and its outputs, as given by the Return statement, are passed as inputs to the adversary. Now the latter executes, oracle queries being answered by the procedures for this purpose associated to the game. The output of the adversary becomes the input to the Finalize procedure of the game. The output of the game is whatever is returned by the Finalize procedure. The boxed games include the boxed statements in the code and the unboxed games do not.

We let "$G_i^A \Rightarrow s$" denote the event that the output of Game G_i when executed with an adversary A is s, and we let "G_i^A sets bad" denote that event that Game G_i when executed with A sets the flag bad.

Both for the games and for the adversary, we adopt the convention that boolean variables are automatically initialized to false and arrays begin everywhere undefined.

Two games G_1, G_2 are called *identical-until-*bad if their code differs only following the setting of flag bad. Note that this is a purely syntactic condition. We use the following lemma from [14].

Lemma 2.4.1. (**Fundamental Lemma of Game-Playing** [14]) Let Games G_1, G_2 be identical-until-bad. Then

$$\Pr\left[\, G_1^A \Rightarrow s \,\right] \;\leq\; \Pr\left[\, G_2^A \Rightarrow s \,\right] + \Pr\left[\, G_1^A \text{ sets } \mathsf{bad} \,\right] .$$

The lemma is quite useful in order to construct "game chains" that bound an adversary's advantage.

CHAPTER III

DETERMINISTIC ENCRYPTION I

In this chapter, we introduce the notion of *deterministic encryption*. Deterministic encryption can be viewed a strengthening of the standard notion of one-way trapdoor functions that addresses the well-known deficiencies of the former as an encryption scheme articulated by Goldwasser and Micali [47], to the extent possible *without* using randomness in the encryption algorithm. Intuitively, deterministic encryption instead draws upon message entropy to achieve security.

3.1 Deterministic Encryption and its Security

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. We say that Π is *deterministic* if $\mathcal{E}$ is deterministic.

Our first task is to formulate a suitable security defintion for deterministic encryption. In the definition that follows, we do not actually require that the encryption scheme be deterministic, but rather this is an important special case for us.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. A *DET-CPA adversary* $A = (A_0, A_1, A_2)$ against Π operates in three stages. First, A_0 gets as input the security parameter and outputs some state information *state*. Next, A_1 gets as input *state* and outputs a vector of messages $\mathbf{x}$ and a "test" string t. (We clarify that A_1 does not update the state information.) Finally, A_2 gets as input the public key pk, a vector of ciphertexts $\mathbf{c}$, and *state*, and ouptuts a "guess" string g. To an encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ and a DET-CPA adversary $A = (A_0, A_1, A_2)$ we associate

```
**Experiment** $\mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k)$:
$b \xleftarrow{\$} \{0,1\}$ ; $state \xleftarrow{\$} A_0(1^k)$
$(\mathbf{x}_0, t_0), (\mathbf{x}_1, t_1) \xleftarrow{\$} A_1(state)$
$\mathbf{c} \xleftarrow{\$} \mathcal{E}(pk, \mathbf{x}_b)$
$g \xleftarrow{\$} A_2(pk, \mathbf{c}, state)$
If $g = t$ return 1 else return 0
```

We require A_1's output to satisfy $|\mathbf{x}_0| = |\mathbf{x}_1|$ and $|\mathbf{x}_0[i]| = |\mathbf{x}_1[i]|$ for all $1 \leq i \leq |\mathbf{x}_0|$. Moreover, we require that $\mathbf{x}_0[i_1] = \mathbf{x}_0[i_2]$ if and only if $\mathbf{x}_1[i_1] = \mathbf{x}_1[i_2]$ for all $1 \leq i_1, i_2 \leq |\mathbf{x}_0|$. (This reflects the fact that deterministic encryption leaks the equality pattern of the plaintexts; in fact, when the encryption scheme is deterministic we may assume without loss of generality that all the $\mathbf{x}_0[i]$ are distinct.) Define the *DET-CPA advantage* of A against Π as

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) = 2 \cdot \Pr\left[\, \mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k) \Rightarrow 1 \,\right] - 1 \; .$$

Equivalently,

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) = \Pr\left[\, \mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k) \Rightarrow 1 \;\middle|\; b = 1 \,\right] - \Pr\left[\, \mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k) \Rightarrow 1 \;\middle|\; b = 0 \,\right] \; .$$

PERMITTED MESSAGE DISTRIBUTIONS. In absence of additional restrictions on the output distribution of A_1, it is clear that the definition is unachievable by deterministic Π. To see this, consider A_1 that outputs that outputs $(0,0)$ with probability $1/2$ and $(1,1)$ with probability $1/2$. Then $A_2(pk, c)$ could return 0 if $\mathcal{E}(pk, 0) = c$ and 1 otherwise, giving A an advantage of $1/2$. This reflects the fact that trial encryption of candidate messages is always a possible attack when encryption is deterministic.

Thus, in applications we consider security relative to a particular *class* of DET-CPA adversaries. In particular, we say that A has *min-entropy* $\mu(\cdot)$ if $\mathrm{H}_\infty(X_i) \geq \mu$ where X_i is the random variable with the distribution of $\mathbf{x}[i]$ over $(\mathbf{x}, t) \xleftarrow{\$} A_1(state)$, for all *state* output by A_0 and all $1 \leq i \leq v$ where A_1 outputs vectors of length v.

Generalizing the above, a necessary condition for A to have advantage at most ε is that its min-entropy is $\log 1/\varepsilon$.

Access to the public key. If A_0 or A_1 were given pk, the definition would again be unachievable for deterministic Π. Indeed, A_1 could output $(\mathbf{x}, t)$ where $\mathbf{x}[1]$ is chosen at random from $\{0,1\}^k$, $|\mathbf{x}| = 1$, and $t = \mathcal{E}(pk, \mathbf{x})$. Then $A_2(1^k, pk, c)$ could return c, and A would have min-entropy k but

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) \;\geq\; 1 - 2^{-k} \; .$$

Intuitively, the ciphertext is non-trivial information about the plaintext, showing that any deterministic scheme leaks information about the plaintext that depends on the public key. Our definition asks that information unrelated to the public key not leak. Note that this also means that we provide security only for messages unrelated to the public key, which is acceptable in practice because normal data is unlikely to depend on any public key. In real life, public keys are abstractions hidden in our software, not strings we look at. However, this does reveal a limitation of deterministic encryption that may not be obvious at first glance.

The state. Note that since A_0 does not take input the public key, we can always hardwire A_1 and A_2 with the "best" state value for A (i.e., which maximizes A's advantage). Thus, we may assume without loss of generality that a given DET-CPA adversary A has "empty" A_0 (meaning A_0 outputs the empty string); we write such an adversary as $A = (A_1, A_2)$ to indicate this.[1] However, allowing non-empty A_0 greatly facilitates some proofs.

Single or multiple messages. The classical definitions explicitly only model the encryption of a single plaintext, but a simple hybrid argument from [4] shows that

[1]Actually, in the random oracle (RO) model [11], where A_0 is given access to the RO (see below), this is unclear because the best state could depend on the values returned by the RO. However, we show in Lemma 3.2.2 that against what we call "public-key respecting" schemes we may assume without loss of generality that A_0 does not make any RO queries. Since all our schemes are public-key respecting, we may thus assume "empty" A_0 even in the RO model.

security when multiple plaintexts are encrypted follows. This hybrid argument fails in our setting, and in fact the two versions are not equivalent, which is why we have explicitly considered the encryption of multiple messages. We discuss this further in Chapter 4 when we consider other definitional equivalences.

Chosen-ciphertext security. For simplicity, the definition of security given here only treats chosen-plaintext security. Extending them to chosen-ciphertext security is straightforward (essentially, A_2 is given a decryption oracle). We further address this notion, which we call DET-CCA, in Chapter 5.

Random oracle model. In the random (RO) oracle model [11], all algorithms, both of the scheme and of the adversary, are given oracle access to the ROs. (In the formal model there is only one RO, but it is easy to see that this is equivalent to multiple ROs.) A security experiment begins by selecting the ROs uniformly at random from the set of all functions with appropriate domain and range. For simplicity, we do not make this selection explicit in the DET-CPA experiment above.

3.2 Some Useful Lemmata

We establish two lemmata that will be useful in the proofs in this section.

Max public-key probability. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme. For all $k \in \mathbb{N}$, define $\mathsf{mpk}_\Pi(k)$ to be the maximum, taken over all $w \in \{0,1\}^*$, of the quantity

$$\Pr\left[\, pk = w \,:\, (pk, sk) \overset{\$}{\leftarrow} \mathcal{K}(1^k) \,\right] .$$

We call $\mathsf{mpk}_\Pi(\cdot)$ the *max public-key probability* of Π. We use the following simple lemma about it, which says that for IND-CPA security the max public-key probability of a scheme must be small.

Lemma 3.2.1. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Then there is an adversary $A = (A_1, A_2)$ such that for all $k \in \mathbb{N}$

$$\mathsf{mpk}_{\Pi}(k) \;\leq\; \sqrt{\mathbf{Adv}^{\text{ind-cpa}}_{\Pi,A}(k)} \;. \tag{2}$$

Furthermore, the running-time of A is at that for $O(1)$ computations of $\mathcal{K}, \mathcal{D}$.

We note that although Lemma 3.2.1 shows that for IND-CPA security the max public-key probability of a scheme must be small, the reduction is not tight. For most specific schemes, one can in fact easily and unconditionally (meaning, without assumption) show that the max public-key probability is small. For example, in ElGamal [43], the public key is g^x, where x is a random exponent in the secret key. In this case, the max public-key probability is $1/|G|$, where $|G|$ is the order of the corresponding group. Using some facts about the density of primes (see [74, Theorem 5.3]), one can show that the max public-key probability of RSA is $O(k/2^k)$.

Proof. Consider the following IND-CPA adversary $A = (A_1, A_2)$:

```
Algorithm $A_1(pk)$:
    Return $(0,1)$

Algorithm $A_2(pk, c)$:
    $(pk', sk') \xleftarrow{\$} \mathcal{K}(1^k)$
    If $pk \neq pk'$ return 0
    $b' \leftarrow \mathcal{D}(1^k, pk', sk', c)$
    Return $b'$
```

Then

$$\Pr\left[\,\mathbf{Exp}^{\text{ind-cpa}}_{\Pi,A}(k) \Rightarrow 1 \;|\; b = 1\right] \;=\; \Pr\left[\, pk = pk' \,\right] \;\geq\; (\mathsf{mpk}_{\Pi}(k))^2$$

and

$$\Pr\left[\,\mathbf{Exp}^{\text{ind-cpa}}_{\Pi,A}(k) \Rightarrow 1 \;|\; b = 0\right] \;=\; 0 \,.$$

Subtracting, we get

$$\mathbf{Adv}^{\text{ind-cpa}}_{\Pi,A}(k) \;\geq\; (\mathsf{mpk}_{\mathsf{AE}})^2 \;,$$

which is equivalent to (2). The claimed running-time of A is easy to verify. $\square$

RO-OBLIVIOUS ADVERSARIES. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a RO-model encryption scheme. We say that a RO query x made by $\mathcal{E}(pk, \cdot)$ or $\mathcal{D}(pk, \cdot)$ is *public-key prefixed* if it has the form $x = pk \| y$ for some string y. We say Π is *public-key respecting* if $\mathcal{K}$ makes no RO queries and every RO query of $\mathcal{E}$ or $\mathcal{D}$ is public-key prefixed. We say that $A = (A_0, A_1, A_2)$ is *RO-oblivious* if A_0, A_1 make no RO queries and A_3 makes only public-key prefixed queries. The following says that for DET-CPA security against public-key respecting schemes it suffices to consider RO-oblivious adversaries. (Note that here we explicitly allow adversaries to have non-empty state, since, as discussed above, in general the "best" state could depend on the values of the RO.)

Lemma 3.2.2. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a RO-model encryption scheme. Let $F = \{F_k\}_{k \in \mathbb{N}}$ be a family of functions with the same domain and range as the RO for Π. If Π is public-key respecting then for any adversary $B = (B_0, B_1, B_2)$ against Π there is a RO-oblivious adversary $A = (A_0, A_1, A_2)$ against Π and a PRF adversary D against F such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) \quad \leq \quad \mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) + 2q \cdot \mathsf{mpk}_{\mathsf{AE}} + \mathbf{Adv}^{\text{prf}}_{F,D}(k) \; ,$$

where q is the maximum number of queries B_1, B_2 make to their RO in total. Furthermore, the running-time of A is the time to run B plus at most q computations of F, and A makes at most q queries to its RO.

We note that pseudorandom functions can be constructed in theory from any one-way function [45], which is trivially implied by a DET-CPA scheme, and so the former does not constitute an additional cryptographic assumption in Lemma 3.2.2.

Proof. Consider the following DET-CPA adversary $A = (A_0, A_1, A_2)$:

```
**Alg** $A_0(1^k)$:
$K \xleftarrow{\$} \mathcal{K}$
Run $B_0$ on $1^k$:
    **On** $H$ **query** $x$:
    Return $F(K,x)$
To recieve $state$
Return $K\|state$
```

```
**Alg** $A_1(K\|state)$:
Run $B_1$ on $state$:
    **On** $H$ **query** $x$:
    Return $F(K,x)$
To recieve $(\mathbf{x},t)$
Return $\mathbf{x},t,K\|state$
```

```
**Alg** $A_2^{H(\cdot)}(pk,\mathbf{c},K\|state)$:
Run $B_2$ on $(pk,\mathbf{c},state)$:
    **On query Hash**$(x)$:
    If $x$ is public-key prefixed
        Return $H(x)$
    Else return $F(K,x)$
To recieve $g$
Return $g$
```

Note that A is RO-oblivious as required. Let BAD be the event that B_0 or B_1 makes a public-key prefixed query to its RO. Then

$$\begin{aligned}\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) &\leq \Pr\left[\mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k) \Rightarrow 1 \mid \overline{\mathsf{BAD}}\right] + \Pr\left[\,\mathsf{BAD}\,\right] \\ &\leq \Pr\left[\mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k) \Rightarrow 1 \mid \overline{\mathsf{BAD}}\right] + 2q\mathsf{mpk}_\Pi(k)\ ,\end{aligned}$$

where the last line is because B is run twice in the DET-CPA experiment and B_1, B_2 get no information about pk. We next claim that for a PRF adversary D that is standard to construct

$$\Pr\left[\mathbf{Exp}^{\text{det-cpa}}_{\Pi,A}(k) \Rightarrow 1 \mid \overline{\mathsf{BAD}}\right] \leq \mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) + \mathbf{Adv}^{\text{prf}}_{\Pi,D}(k)\ .$$

This follows by a standard hybrid argument, and then observing that if BAD does not occur then B's view is the same as in the DET-CPA experiment. Substituting, we get (3). The claimed resource usage of the constructed adversaries is easy to verify. □

3.3 Constructions in the Random Oracle Model

Here we present deterministic encryption schemes meeting our new notion in the random oracle model. Specifically, we present schemes achieving

- DET-CPA security based on any IND-CPA (probabilistic) encryption scheme (in fact, as we show in Chapter 5, the scheme is DET-*CCA* secure).

- DET-CPA security based on RSA, where furthermore the scheme is length-preserving.

These constructions meet security under the minimal requirements on permitted message distributions for deterministic encryption as discussed in Section 3.1.

3.3.1 The Encrypt-with-Hash Scheme

The scheme. We show a generic construction of DET-CPA deterministic encryption scheme from any IND-CPA randomized one. Our construction replaces the coins used by the latter with the hash of the message and the public key. More formally, let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Say that $\mathcal{E}(pk, x)$ draws its coins from a set $\mathsf{Coins}_{pk}(|x|)$. We assume the RO has the property that $H(pk\|x) \in \mathsf{Coins}_{pk}(|x|)$ for all pk output by $\mathcal{K}$ and all $x \in \{0,1\}^*$. Define the RO-model *Encrypt-with-Hash* deterministic encryption scheme $\mathsf{EwHash}[\Pi] = (\mathcal{K}, \mathcal{DE}, \mathcal{DD})$ associated to Π via

```
Algorithm $\mathcal{DE}^{H(\cdot)}(pk, x)$:
  $r \leftarrow H(pk\|x)$
  $c \leftarrow \mathcal{E}(pk, x; r)$
  Return $c$

Algorithm $\mathcal{DD}^{H(\cdot)}(sk, c)$:
  $x \leftarrow \mathcal{D}(sk, c)$
  $r \leftarrow H(pk\|x)$
  If $\mathcal{E}(pk, x; r) = c$ then return $x$
  Else return $\perp$
```

Security analysis. The security proof for the Encrypt-with-Hash scheme is more subtle than it may appear. Intuitively, we would like to say that, unless the adversary queries the high-entropy message to its hash oracle, then the hash value looks just like random coins. But for the former to hold we need to appeal to IND-CPA security of the starting scheme (to argue that the adversary does not get any additional information about the message), which itself requires coins to be random. The security proof resolves this apparent circularity.

Theorem 3.3.1. Let $A = (A_1, A_2)$ be a RO-oblivious DET-CPA adversary against $\mathsf{EwHash}[\Pi]$ with min-entropy μ, which outputs vectors of length at most v and makes at most q queries to its hash oracle. Then there is an IND-CPA adversary $B = (B_1, B_2)$ against Π such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathsf{det\text{-}cpa}}_{\mathsf{EwHash},A}(k) \;\leq\; \mathbf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,B}(k) + \frac{qv}{2^{\mu-1}} \;. \tag{3}$$

Furthermore, the running-time of B is at most that of A plus $O(vn)$.

Proof. Adversary B is given in Figure 3. Consider the games depicted in Figure 4. We claim the following sequence of inequalities:

$$\begin{aligned}
\Pr\left[\, G_1^{A_2} \Rightarrow b \,\right] &\leq \Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] && (4)\\
&\leq \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \Pr[G_2 A_2 \text{ sets } \mathsf{bad}\,] && (5)\\
&\leq \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \frac{vq}{2^{-\mu}} && (6)\\
&= \Pr\left[\, G_4^{A_2} \Rightarrow b \,\right] + \frac{vq}{2^{-\mu}} && (7)\\
&= \Pr\left[\, G_5^{A_2} \Rightarrow b \,\right] + \frac{vq}{2^{-\mu}} && (8)
\end{aligned}$$

by which Equation (3) follows from multiplying each side by 2 and subtracting 1, taking into account the definition of the advantages of A, B.

To see Equation 4, observe that the Finalize procedure of Game G_2 begins by defining its output bit d in certain ways depending on the flags $\mathsf{zer}, \mathsf{one}$ if either of these are true, and otherwise defining it as in G_1. However, in case the value of d set by the first two “If” statements is wrong, meaning not equal to b, the third “If” statement corrects, setting d to b. The net result is that in the cases that G_2 assigns d differently from G_1, the assignment made by G_2 is correct, meaning equal to b. Additionally G_2 sets a flag bad but this does not influence its choice of d. So the probability that d equals b can only go up.

As Games G_2, G_3 identical-until-bad, Lemma 2.4.1 applies to justify (5). The probability that A_2 makes hash query $x \in \mathbf{x}_{1-b}$ when executed with G_3 is at most $qv/2^{-\mu}$ by a union bound because A_2 gets no information about $\mathbf{x}_{1-b}$. This justifies (6). Since the third “If” statement in G_3 only sets a flag that does not influence the

```
**Adversary** $B_1(pk)$:
$(\mathbf{x}_0, t_0), (\mathbf{x}_1, t_1) \xleftarrow{\$} A_1(1^k)$
$state \leftarrow \mathbf{x}_0 \| t_0 \| \mathbf{x}_1 \| t_1$
Return $(\mathbf{x}_0, \mathbf{x}_1, state)$
```

```
**Adversary** $B_2(pk, \mathbf{c}, state)$:
$\mathbf{x}_0 \| t_0 \| \mathbf{x}_1 \| t_1 \leftarrow state$
Run $A_2$ on inputs $pk, \mathbf{c}$:
    **On hash query** $pk \| x$ do:
    If $H[x]$ is undefined then
        $H[x] \xleftarrow{\$} \mathsf{Coins}_{pk}(|x|)$
        If $x \in \mathbf{x}_0$ then
            If $\mathsf{one} = \mathsf{false}$ then $\mathsf{zer} \leftarrow \mathsf{true}$
        If $x \in \mathbf{x}_1$ then
            If $\mathsf{zer} = \mathsf{false}$ then $\mathsf{one} \leftarrow \mathsf{true}$
    Return $H[x]$
Let $g$ be the output of $A_2$
If $\mathsf{zer} = \mathsf{true}$ then $d \leftarrow 0$
Else If $\mathsf{one} = \mathsf{true}$ then $d \leftarrow 1$
      Else If $g = t_1$ then $d \leftarrow 1$ else $d \leftarrow 0$
Return $d$
```

Figure 3: IND-CPA adversary B for proof of Theorem 5.5.1.

game output, dropping this entire statement results in an equivalent game G_4. This justifies Equation (6).

To see Equation 7, we can consider a common finite space of coins associated to the executions of A_2 with either G_4 or G_5. Consider the execution of A_2 with G_4 when a particular coin sequence is chosen at random from this set. One of the boxed statements in the procedure to respond to a hash query can be executed only if either $\mathsf{one} = \mathsf{true}$ or $\mathsf{zer} = \mathsf{true}$, due to the "If" statements that precede the boxed statements. However, once one of these flags is set to true, the output of the Finalize procedure is determined. (Nothing further that happens in the execution can change it. Note we use here that at most one of $\mathsf{zer}, \mathsf{one}$ can be true, never both, and once one of them is true, it never becomes false.) This means that the boxed statements have no effect on the output of the game, and eliminating them results in the equivalent game G_5. □

```
procedure Initialize: All games
$b \xleftarrow{\$} \{0,1\}$
$(\mathbf{x}_0, t_0), (\mathbf{x}_1, t_1) \xleftarrow{\$} A_1(1^k)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
For $i \in \{0,1\}, j \in [v]$ do:
    $R_{i,j} \xleftarrow{\$} \mathsf{Coins}_{pk}(|\mathbf{x}_i[j]|)$
    $\mathbf{c}_i[j] \leftarrow \mathcal{E}(pk, \mathbf{x}_i[j]; R_{i,j}$
Return $pk, \mathbf{c}$
```

```
On H query $pk \| x$: $\boxed{G_1\text{–}G_4}, G_5$
If $H[x]$ is undefined then
    $H[x] \xleftarrow{\$} \mathsf{Coins}_{pk}(|x|)$
If $\exists i$ such that $x = \mathbf{x}_0[i]$ then
    If $\mathsf{one} = \mathsf{false}$ then $\mathsf{zer} \leftarrow \mathsf{true}$
    $\boxed{H[x] \leftarrow R_{0,i}}$
If $\exists i$ such that $x = \mathbf{x}_1[i]$ then
    If $\mathsf{zer} = \mathsf{false}$ then $\mathsf{one} \leftarrow \mathsf{true}$
    $\boxed{H[x] \leftarrow R_{1,i}}$
Return $H[x]$
```

```
procedure Finalize($g$): $G_1$
If $g = t_1$ then $d \leftarrow 1$ else $d \leftarrow 0$
Return $d$
```

```
procedure Finalize($g$): $G_2, \boxed{G_3}$
If $\mathsf{zer} = \mathsf{true}$ then $d \leftarrow 0$
Else If $\mathsf{one} = \mathsf{true}$ then $d \leftarrow 1$
Else If $g = t_1$ then $d \leftarrow 1$
Else $d \leftarrow 0$
If $(b = 1 \wedge \mathsf{zer} = \mathsf{true})$
    $\vee\ (b = 0 \wedge \mathsf{one} = \mathsf{true})$
    then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow b}$
Return $d$
```

```
procedure Finalize($g$): $G_4, G_5$
If $\mathsf{zer} = \mathsf{true}$ then $d \leftarrow 0$
Else If $\mathsf{one} = \mathsf{true}$ then $d \leftarrow 1$
Else If $g = t_1$ then $d \leftarrow 1$
Else $d \leftarrow 0$
Return $d$
```

Figure 4: Games for the proof of Theorem 3.3.1. The boxed labels indicate which games include the boxed statements and which do not.

3.3.2 The RSA-DOAEP Scheme

It is sometimes important to minimize the number of bits transmitted over the network. We devise an efficient deterministic encryption scheme that is optimal in this regard, namely is length-preserving. (That is, the length of the ciphertext equals the length of the plaintext.) Length-preserving schemes can also be needed for securing legacy code. Our construction is inspired by RSA-OAEP [12]. But in place of the randomness in this scheme we use part of the message, and in contrast to RSA-OAEP our scheme requires a 3-round rather than 2-round underlying Feistel transform.

The scheme. Formally, the scheme is parameterized by integers $k_0, k_1 > 0$. The plaintext space PtSp(k) consists of all strings of length at least $\max(k_1, 2k_0 + 1)$.

We assume here for simplicity that all messages to encrypt have a fixed length $n = n(k)$ satisfying $n > 2k_0$ and $n \geq k_1$. Let $\mathcal{RSA}$ be the RSA trapdoor-permutation generator with modulus length k_1. The key-generation algorithm of the associated RO-model deterministic encryption scheme $\mathsf{RSA\text{-}DOAEP}$ ("D" for "deterministic") is $\mathcal{RSA}$. The encryption and decryption algorithms have oracle access to functions $H_1, H_2 \colon \{0,1\}^* \to \{0,1\}^{n-k_0}$ and $R \colon \{0,1\}^* \to \{0,1\}^{k_0}$, and are defined as follows:

```
Algorithm $\mathcal{E}^{H_1,H_2,R}((N,e),x)$:
$x_l \leftarrow x[1 \dots k_0]$
$x_r \leftarrow x[k_0+1 \dots n]$
$s_0 \leftarrow H_1((N,e)\|x_r) \oplus x_l$
$t_0 \leftarrow R((N,e)\|s_0) \oplus x_r$
$s_1 \leftarrow H_2((N,e)\|t_0) \oplus s_0$
$x_1 \leftarrow (s_1\|t_0)[1 \dots n-k_1]$
$x_2 \leftarrow (s_1\|t_0)[n-k_1+1 \dots n]$
$y \leftarrow x_1\|(x_2^e \bmod N)$
Return $y$
```

```
Algorithm $\mathcal{D}^{H_1,H_2,R}((N,e,d),y)$:
$x_1 \leftarrow y[1 \dots n-k_1]$
$y_1 \leftarrow y[n-k_1+1 \dots n]$
$x \leftarrow x_1\|(y_1^d \bmod N)$
$s_1 \leftarrow x[1 \dots k_0]$
$t_0 \leftarrow x[k_0+1 \dots n]$
$s_0 \leftarrow H_2((N,e)\|t_0) \oplus s_1$
$x_r \leftarrow R((N,e)\|s_0) \oplus t_0$
$x_l \leftarrow H_1((N,e)\|x_r) \oplus s_0$
Return $x_l\|x_r$
```

Security analysis. We prove that RSA-DOAEP is secure in our sense assuming RSA is one-way.

Theorem 3.3.2. Let $A = (A_1, A_2)$ be a RO-oblivious DET-CPA adversary against $\mathsf{RSA\text{-}DOAEP}$ with min-entropy μ that makes at most $q_{\mathrm{h_i}}$ queries to oracle H_i for $i \in \{1,2\}$ and q_{r} to oracle R, and outputs vectors of size v with components of length n. We consider two cases:

- Case 1: $n < k_0 + k_1$. Then there is an inverter I against $\mathcal{RSA}$ such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\mathsf{RSA\text{-}DOAEP},A}(k) \leq q_{\mathrm{h}_2} v \cdot \sqrt{\mathbf{Adv}^{\text{owf}}_{\mathcal{RSA},I}(k) + 2^{2k_1-4(n-k_0)+5}} + \quad (9)$$

$$\frac{2q_{\mathrm{r}}v}{2^{k_0}} + \frac{2q_{\mathrm{h}_1}q_{\mathrm{r}}v}{2^{\mu}} \, . \quad (10)$$

Furthermore the running-time of I is at most twice that of A plus $O(\log v + q_{\mathrm{h}_2} \log q_{\mathrm{h}_2} + k_1^3)$.

- Case 2: $n \geq k_0 + k_1$. Then there is an inverter I against $\mathcal{RSA}$ such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\mathsf{RSA\text{-}DOAEP},A}(k) \;\leq\; v \cdot \mathbf{Adv}^{\text{owf}}_{\mathcal{RSA},I} + \frac{2q_{\mathrm{r}}v}{2^{k_0}} + \frac{2q_{\mathrm{h}_1}q_{\mathrm{r}}v}{2^{\mu}} \ .$$

  Furthermore, the running-time of I is at most that of A plus $O(\log v + q_{\mathrm{h}_2} \log q_{\mathrm{h}_2})$.

In practice, we will have, e.g. $k_1 = 2048$, and then we can set parameter k_0 to, say, 160 bits to effectively maximize security in either case of the theorem. Then, the relation between $n - 160$ and 2048 then determines which case of the theorem applies.

Proof. We prove Case 1, meaning we assume that $n - k_0 < k_1$. For the proof, we construct a partial one-way adversary against $\mathcal{RSA}$ and then conclude by Lemma 2.3.2. The former, which we call $\mathsf{GetQuery}$, is given in Figure 5. The games for the proof are in Figure 6, Figure 7, and Figure 8. Equation (10) follows from the following sequence of inequalities, which we will justify below:

$$
\begin{aligned}
\Pr\left[\, G_1^{A_2} \Rightarrow b \,\right] &= \Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] + \Pr[G_1^{A_2} \text{ sets } \mathsf{bad}_1\,] && (11)\\
&\leq \Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} && (12)\\
&\leq \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \Pr[G_2^{A_2} \text{ sets } \mathsf{bad}_2\,] && (13)\\
&\leq \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} && (14)\\
&\leq \Pr\left[\, G_4^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} + \Pr[G_3^{A_2} \text{ sets } \mathsf{bad}_3\,] \\
&\leq \Pr\left[\, G_5^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} + \Pr[G_3^{A_2} \text{ sets } \mathsf{bad}_3\,] \\
&\quad + \Pr[G_4^{A_2} \text{ sets } \mathsf{bad}_4\,] && (15)\\
&= \Pr\left[\, G_6^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} + \Pr[G_3^{A_2} \text{ sets } \mathsf{bad}_3\,] \\
&\quad + \Pr[G_4^{A_2} \text{ sets } \mathsf{bad}_4\,] && (16)\\
&\leq \Pr\left[\, G_7^{A_2} \Rightarrow b \,\right] + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} + \Pr[G_3^{A_2} \text{ sets } \mathsf{bad}_3\,] \\
&\quad + \Pr[G_4^{A_2} \text{ sets } \mathsf{bad}_4\,] + \Pr[G_6^{A_2} \text{ sets } \mathsf{bad}_5\,] && (17)\\
&\leq \frac{1}{2} + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} + \Pr[G_3^{A_2} \text{ sets } \mathsf{bad}_3\,] \\
&\quad + \Pr[G_4^{A_2} \text{ sets } \mathsf{bad}_4\,] + \Pr[G_5^{A_2} \text{ sets } \mathsf{bad}_5\,] && (18)\\
&\leq \frac{1}{2} + \mathbf{Adv}^{\mathrm{powf}}_{\mathcal{RSA},\mathsf{GetQuery}}(k) + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}} && (19)\\
&\leq \frac{1}{2} + q_{\mathrm{h}_2} v \cdot \sqrt{\mathbf{Adv}^{\mathrm{owf}}_{\mathcal{RSA},I}(k) + 2^{2k_1 - 4(n-k_0)+10}} \\
&\quad + 2^{k_1 - 2(n-k_0)+5} + \frac{q_{\mathrm{r}} v}{2^{k_0}} + \frac{q_{\mathrm{h}_1} q_{\mathrm{r}} v}{2^{\mu}}\,. && (20)
\end{aligned}
$$

Lemma 2.4.1 applies to justify (11). To bound the probability that Game G_1 when executing A_2 sets bad_1, note that without A_2 querying $(N,e)\|m_{i,r}$ (by which we mean for some $i \in \{1,\dots v\}$) to H_1 nor $(N,e)\|T_{i,0}$ to H_2, the values of $S_{i,1}\|T_{i,0}, R_i^*$ are random and independently distributed of $\mathbf{x}_b$ from its perspective. To make this more precise, we define an auxilliary game called G_{rand} in which all input to A_2 and answers to its oracle queries are independent and random of $\mathbf{x}_b$. We claim that the probability G_1 when executing A_2 sets bad_1 is the same that G_{rand} does. To see this, consider a common finite space of coins associated to the executions of A_2 in either

G_1 or G_{rand}. If A_2 when executed using a particular sequence of coins from this space causes G_1 to set bad_1 with some probability then A_2 when executed using this same coin sequence also causes G_{rand} to set bad_1 with the same probability, because on such a coin sequence the input to A_2 and the game's oracle replies are identically distributed from the perspective of A_2 up to the point that bad_1 is set. Since when executed in G_{rand}, A_2 gets no information about $H^*_{1,i}$ for any $1 \leq i \leq v$, the probability that G_{rand} when executing A_2 sets bad_1 is at most $q_{\text{r}}v/2^{k_0}$, giving Equation (12).

Equation (13) is again obtained via Lemma 2.4.1. We bound the probability that G_2 when executing A_2 sets bad_2 as follows. This is the same as the probability that G_{rand} does, by an analogous argument to the above, considering the fact that without having queried $(N,e)\|S_{i,0}$ to R nor $(N,e)\|T_{i,0}$ to H_2, the value of $S_{i,1}\|T_{i,0}$ is random and independently distributed of $\mathbf{x}_b$ from the perspective of A_2. The probability of that G_{rand} sets bad_2 is, in turn, the same, over a common finite set of coins with which A_2 is executed, as the probability that the "knowledge extractor" K given in Figure 9 outputs a list containing the plaintext $\mathbf{x}_b[i]$ for some $1 \leq i \leq v$. The probability that K outputs such a list is at most $q_{\text{h}_1}q_{\text{r}}v/2^{\mu}$ because it gives A_2 no information about $\mathbf{x}_b$, giving (14). (Here is where we use the fact that the padding transform of RSA-DOAEP consists of *three* Feistel rounds; with only two, this step does not go through.)

Lemma 2.4.1 applies to justify all of (15), (15), and (17). We delay bounding the probabilities here until later.

Next consider when A_2 executed with Game G_5 queries $(N,e)\|m_{i,r}$ to H_1 but prior to this has queried neither $(N,e)\|S_{i,0}$ to R nor $(N,e)\|T_{i,0}$ to H_2. Then, in reply to query $(N,e)\|m_{i,r}$, it receives $H^*_{i,1}$, which is random and independent of everything given to A_2 so far. Then, after it queries $(N,e)\|m_{i,r}$ to H_1, we see from the code that the answers given to A_2 in reply to any of its queries are likewise random and independent. This means that, instead replying to query $(N,e)\|m_{i,r}$ with the special string $H^*_{i,1}$ defined at the beginning of the game, we could simply reply with a random and independent string chosen "on the fly" during the particular invocation of the procedure to respond to H_1 queries. In other words, we may drop the "Else" statement in this procedure to result in an equivalent game G_5, which justifies (16).

Now, we have that the probability that G_7 outputs the challenge bit b chosen randomly at the beginning of the game when executing A_2 is at most $1/2$, because this game does not give A_2 any information about $\mathbf{x}_b$, giving (18).

Finally, observe that in each of the following cases, A_2 when executed with a particular sequence of coins (drawn, as usual, from a common finite set associated to the execution of A_2 in either game) causing the relevant game to set the flag also causes G_{rand} to do the same: game G_3 sets bad_3, game G_4 set bad_4, and game G_6 sets bad_5. This is because on such a coin sequence the input to A_2 and its oracle replies have the same distribution (from the perspective of A_2) in either game. In the first case, this can be seen directly from the code. In the second case, observe that in game G_5 until A_2 queries $(N,e)\|T_{i,0}$ to H_2 then on query $(N,e)\|m_{i,r}$ H_1 the response it receives, namely $H^*_{i,2}$ defined at the beginning of the game, is in fact random and independent of everything given to by the game. In the third case, this is again clear from the code. Moreover, these cases exhaust all the possible sequences of queries made by A_2 for which A_2 queries $(N,e)\|T_{i,0}$ for some i to its H_2 oracle. Now the input to A_2 and its oracle replies are distributed identically when executed with G_{rand} and when run by algorithm $\mathsf{GetQuery}$, except that the procedure to respond to queries to H_2 in G_{rand} explicitly checks whether a query made by A_2 is equal to $T_{i,0}$ for some i, whereas algorithm $\mathsf{GetQuery}$ simply guesses whether this is the case by picking j when the first such query (which it hopes is $T_{w,0}$) will occur, at which point its output is determined. (Game G_{rand} also makes some "If" checks omitted by $\mathsf{GetQuery}$ and sets some flags, but these do not influence game output.) Since w, j are random and independent and A_2 gets no information about them, we have

$$\begin{aligned}\Pr[G_2^{A_2} \text{ sets } \mathsf{bad}_2\,] \;&+\; \Pr[G_4^{A_2} \text{ sets } \mathsf{bad}_3\,] + \Pr[G_6^{A_2} \text{ sets } \mathsf{bad}_4\,] \\ &\leq\; q_{\mathrm{h}_2} v \cdot \mathbf{Adv}^{\mathrm{powf}}_{\mathcal{RSA},\mathsf{GetQuery}}(k)\end{aligned}$$

justifying (19). Equation (20) then follows by Lemma 2.3.2. Multiplying both sides by 2 then subtracting 1 and taking into account the definition of the advantage of A yields Equation 10.

To finish the proof, we justify the running-time analysis of I by taking into account the convention that the running-time of A includes that of its overlying experiment. Additional time-complexity for $\mathsf{GetQuery}$ here is for picking two random numbers between 1 and q_{h_2}, v, respectively, and maintaining a counter up to value at most q_{h_2}, incremented each time A_2 makes a query to oracle H_2, which is $O(\log v + q_{\mathrm{h}_2} \log q_{\mathrm{h}_2})$. Then applying the running-time analysis in Lemma 2.3.2, we have that the running-time of I twice that of $\mathsf{GetQuery}$ plus $O(k_1^3)$ as claimed. $\square$

```
**Algorithm** GetQuery$((N,e),y)$
$ctr \leftarrow 0$
$j \xleftarrow{\$} \{1,\ldots,q_{H_2}\}$ ; $w \xleftarrow{\$} \{1,\ldots,v\}$
$\mathbf{c} \xleftarrow{\$} (\{0,1\}^n)^{\times v}$   /* pick random $v$-size vector */
$y' \xleftarrow{\$} \{0,1\}^{n-k_1}$
$\mathbf{c}[w] \leftarrow y' \| y$
Run $A_2$ on inputs $(N,e), \mathbf{c}$:
    **On** $H_1$ **query** $((N,e)\|x$
    If $H_1[x]$ is undefined then
        $H_1[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
    Return $H_1[x]$
    **On** $R$ **query** $(N,e)\|x$
    If $R[x]$ is undefined then
        $R[x] \xleftarrow{\$} \{0,1\}^{k_0}$
    Return $R[x]$
    **On** $H_2$ **query** $(N,e)\|x$
    $ctr \leftarrow ctr + 1$
    If $H_2[x]$ is undefined then
        $H_2[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
    If $ctr = j$ then
        $T \leftarrow x$
    Return $H_2[x]$
Until $A_2$ halts
Return $T$
```

Figure 5: Algorithm GetQuery for the proof of Theorem 3.3.2.

```
**procedure Initialize**: All Games
$b \xleftarrow{\$} \{0,1\}$
$(\mathbf{x}_0, t_0), (x_1, t_1) \xleftarrow{\$} A_1(1^k)$
$((N,e),(N,d)) \xleftarrow{\$} \mathcal{RSA}(1^k)$
For $i = 1$ to $v$ do:
    $m_{i,l} \leftarrow \mathbf{x}_b[i][1 \dots k_0]$
    $m_{i,r} \leftarrow \mathbf{x}_b[i][k_0+1 \dots n]$
    $H^*_{i,1}, H^*_{i,2} \xleftarrow{\$} \{0,1\}^{n-k_0}$ ; $R^*_i \xleftarrow{\$} \{0,1\}^{k_0}$
    $S_{i,0} \leftarrow H^*_{i,1} \oplus m_{i,l}$ ; $T_{i,0} \leftarrow R^*_i \oplus m_{i,r}$
    $S_{i,1} \leftarrow H^*_{i,2} \oplus S_{i,0}$
    $y_{i,l} \leftarrow (S_{i,1} \| T_{i,0})[1 \dots n-k_1]$
    $y_{i,r} \leftarrow ((S_{i,1} \| T_{i,0})[n-k_1+1 \dots n])^e$
    $\mathbf{c}[i] \leftarrow y_{i,l} \| y_{i,r}$
Return $((N,e), \mathbf{c})$
```

```
**On $H_1$ query** $(N,e)\|x$: $\boxed{G_1}$, $G_2$
If $H_1[x]$ is undefined then
    $H_1[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
If $\exists i$ such that $x = m_{i,r}$ then
    If $H_{i,2}[T_{i,0}]$ is defined then
        $H_1[x] \leftarrow H^*_{i,1}$
    If $R[S_{i,0}]$ is defined then
        $\boxed{H_1[x] \leftarrow H^*_{i,1}}$ Return $H_1[x]$
```

```
**On $R$ query** $(N,e)\|x$: $\boxed{G_1}$, $G_2$
If $R[x]$ is undefined then
    $R[x] \xleftarrow{\$} \{0,1\}^{k_0}$
If $\exists i$ such that $x = S_{i,0}$ then
    If $H_2[T_{i,0}]$ is defined then
        $R[x] \leftarrow R^*_i$
        If $H_1[m_r]$ is undefined then
            $\mathsf{bad}_1 \leftarrow \mathsf{true}$ ; $\boxed{R[x] \leftarrow R^*_i}$
Return $R[x]$
```

```
**On $R$ query** $(N,e)\|x$: $\boxed{G_2}$, $G_3$
If $R[x]$ is undefined then
    $R[x] \xleftarrow{\$} \{0,1\}^{k_0}$
If $\exists i$ such that $x = S_{i,0}$ then
    If $H_2[T_{i,0}]$ is defined then
        $R[x] \leftarrow R^*_i$
    If $H_1[m_{i,r}]$ is defined then
        $\mathsf{bad}_2 \leftarrow \mathsf{true}$ ; $\boxed{R[x] \leftarrow R^*_i}$
Return $R[x]$
```

Figure 6: Games for the proof of Theorem 3.3.2. The boxed labels indicate which games include the boxed statements and which do not. All games have the same Finalize procedure, namely **procedure Finalize**(g): If $g = t_0$ then Return 1, Else Return 0.

```
On $H_2$ query $(N,e)\|x$: $\boxed{G_1 - G_3}$, $G_4$
If $H_2[x]$ is undefined then
    $H_2[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
If $\exists i$ such that $x = T_{i,0}$ then
    If $R[S_{i,0}]$ is defined then
        $\mathsf{bad}_3 \leftarrow \mathsf{true}$ ; $\boxed{H_2[x] \leftarrow H^*_{i,2}}$
    Else $H_2[x] \leftarrow H^*_{i,2}$
Return $H_2[x]$
```

```
On $H_2$ query $(N,e)\|x$: $\boxed{G_4}$, $G_5$
If $H_2[x]$ is undefined then
    $H_2[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
If $\exists i$ such that $x = T_{i,0}$ then
    If $H_1[m_{i,r}]$ is defined then
        $\mathsf{bad}_4 \leftarrow \mathsf{true}$ ; $\boxed{H_2[x] \leftarrow H^*_{i,2}}$
    Else $H_2[x] \leftarrow H^*_{i,2}$
Return $H_2[x]$
```

```
On $H_1$ query $(N,e)\|x$:
                    $\boxed{G_5}$, $G_6, G_7$
If $H_1[x]$ is undefined then
    $H_1[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
If $\exists i$ such that $x = m_{i,r}$ then
    If $H_{i,2}[T_{i,0}]$ is defined then
        $H_1[x] \leftarrow H^*_{i,1}$
    Else $\boxed{H_1[x] \leftarrow H^*_{i,1}}$
Return $H_1[x]$
```

```
On $H_2$ query $(N,e)\|x$:
                    $\boxed{G_6}$, $G_7$
If $H_2[x]$ is undefined then
    $H_2[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
If $\exists i$ such that $x = T_{i,0}$ then
    If $R[S_{i,0}], H_1[s\|m_{i,r}]$
        are undefined then
        $\mathsf{bad}_5 \leftarrow \mathsf{true}$ ; $\boxed{H_2[x] \leftarrow H^*_{i,2}}$
Return $R[x]$
```

Figure 7: More games for the proof of Theorem 3.3.2. The boxed labels indicate which games include the boxed statements and which do not. All games have the same Finalize procedure, namely **procedure Finalize**(g): If $g = t_0$ then Return 1, Else Return 0.

```
procedure Initialize:
$b \xleftarrow{\$} \{0,1\}$
$(\mathbf{x}_0, t_1), (\mathbf{x}_1, t_1) \xleftarrow{\$} A_1(1^k)$
$(N,e), (N,d) \xleftarrow{\$} \mathcal{RSA}$
$\mathbf{c} \xleftarrow{\$} (\{0,1\}^n)^{\times v}$
For $i = 1$ to $v$ do :
    $H^*_{i,1}, H^*_{i,2} \xleftarrow{\$} \{0,1\}^{n-k_0}$ ; $R^*_i \xleftarrow{\$} \{0,1\}^k$
    $S_{i,0}, S_{i,1} \xleftarrow{\$} \{0,1\}^{n-k_0}$ ; $T_{i,0} \xleftarrow{\$} \{0,1\}^{k_0}$
Return $((N,e), \mathbf{c})$
```

```
On $H_1$ query $(N,e)\|x$:
If $H_1[x]$ is undefined then
    $H_1[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
Return $H_1[x]$
```

```
On $R$ query $(N,e)\|x$:
If $R[x]$ is undefined then
    $R[x] \xleftarrow{\$} \{0,1\}^{k_0}$
If $\exists i$ such that $x = S_{i,0}$ then
    If $H_1[m_r]$ is undefined then
        $\mathsf{bad}_1 \leftarrow \mathsf{true}$
    Else $\mathsf{bad}_2 \leftarrow \mathsf{true}$
Return $R[x]$
```

```
On $H_2$ query $(N,e)\|x$:
If $H_2[x]$ is undefined then
    $H_2[x] \xleftarrow{\$} \{0,1\}^{n-k_0}$
If $\exists i$ such that $x = T_{i,0}$ then
    If $R[S_{i,0}]$ is defined
        $\mathsf{bad}_3 \leftarrow \mathsf{true}$ ;
    Else If $H_1[m_{i,r}]$ is defined then
        $\mathsf{bad}_4 \leftarrow \mathsf{true}$
    Else $\mathsf{bad}_5 \leftarrow \mathsf{true}$
Return $H_2[x]$
```

```
procedure Finalize($g$):
If $g = t_0$ then Return 1
Else Return 0
```

Figure 8: Game G_{rand} for the proof of Theorem 3.3.2.

```
**Algorithm** $\mathsf{K}(1^k)$
$b \overset{\$}{\leftarrow} \{0,1\}$
$(\mathbf{x}_0, t_0), (\mathbf{x}_1, t_1) \overset{\$}{\leftarrow} A_1(1^k)$
$((N,e),(N,d)) \overset{\$}{\leftarrow} \mathcal{RSA}(1^k)$
$\mathbf{c} \overset{\$}{\leftarrow} (\{0,1\}^n)^{\times v}$    /* pick random $v$-size vector */
Run $A_2$ on input $1^k, (N,e), \mathbf{c}$, replying to its oracle queries as follows:
    **On** $H_1$ **query** $(N,e)\|x$
    If $H_1[x]$ is undefined then
        $H_1[s\|x] \overset{\$}{\leftarrow} \{0,1\}^{n-k_0}$
    Return $H_1[s\|x]$
    **On** $R$ **query** $(N,e)\|x$
    If $R[s\|x]$ is undefined then
        For all $z \in L_1$ add $x \oplus H_1[(N,e)\|z]\|z$ to $L_2$
        $R[x] \overset{\$}{\leftarrow} \{0,1\}^{k_0}$
    Return $R[x]$
    **On** $H_2$ **query** $(N,e)\|x$
    If $H_2[x]$ is undefined then
        $H_2[x] \overset{\$}{\leftarrow} \{0,1\}^{n-k_0}$
    Return $H_2[x]$
Until $A_2$ halts
Return $L_2$
```

Figure 9: Knowledge extractor K for the proof of Theorem 3.3.2.

CHAPTER IV

DETERMINISTIC ENCRYPTION II

In this chapter, we firm up the foundations of deterministic encryption by studying alternative security definitions and constructions without random oracles.

4.1 Definitional Equivalences

The DET-CPA notion captures our intuition about privacy for determinsitic encryption well but is difficult to work with. We would like to find simpler but equivalent definitions for this purpose.

4.1.1 Single Versus Multiple Messages

A basic question is whether there is an equivalence between DET-CPA security for single and multiple messages. That is, we are asking whether in the DET-CPA notion it suffices to consider adversaries that output vectors of size 1. Indeed, classical definitions explicitly only model the encryption of a single plaintext, but a simple hybrid argument from [4] shows that security when multiple plaintexts are encrypted follows. This hybrid argument fails in our setting, and in fact the two versions are not equivalent.

To show this, let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be deterministic encryption scheme. Define $\overline{\Pi} = (\mathcal{K}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ as follows:

```
Algorithm $\overline{\mathcal{E}}(pk, x)$
    $y \leftarrow \mathcal{E}(pk, x)$
    $z \leftarrow \mathcal{E}(pk, \overline{x})$
    Return $y \| z$

Algorithm $\overline{\mathcal{D}}(pk, y \| z)$
    $x \leftarrow \mathcal{D}(sk, y)$
    $x' \leftarrow \mathcal{D}(sk, z)$
    If $x' = \overline{x}$ then return $x$
    Else return $\perp$
```

Above and in what follows, $\overline{s}$ denotes the bitwise complement of a string s. If Π is DET-CPA then $\overline{\Pi}$ is DET-CPA against one-message adversaries. However, the following attack shows $\overline{\Pi}$ is insecure against adversaries that output vectors of size 2. Consider $A = (A_1, A_2)$ where $A_1(1^k)$ picks m_1, m_2 from $\{0,1\}^k$ and a bit b at random, and if $b = 1$ outputs $((m_1, \overline{m}_1), 1)$ and otherwise $((m_1, m_2), 0)$. $A_2(pk, (y_1 \| z_1, y_2 \| z_2))$ outputs 1 if $z_1 = y_2$ and 0 otherwise. Then A has min-entropy k but advantage 1/2.

As a consequence of this, we must, in general, explicitly consider DET-CPA adversaries that output arbitrary plaintext vectors in our security analyses. However, for some specific classes of DET-CPA adversaries (defined by the distributions they put on $\mathbf{x}$) we can recover the equivalence. In particular, Fehr [41] has shown an equivalence in the case of *block-sources* [27], where each plaintext has sufficient entropy conditioned on the outcomes of the previous ones, under the condition that the distribution is "conditionally resampleable" for each block (or for message distributions that are not necessarily efficiently sampleable).

4.1.2 An Indistinguishability-Based Notion

Based on prior work on the encryption of high-entropy messages in the symmetric, information-theoretic context by Dodis and Smith [38], we propose the following simpler indistinguishability-based notion for deterministic encryption that we call "distribution hiding." In particular, this definition will give us a handle on how to achieve deterministic encryption without random oracles in Section 4.2.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. A *DH-CPA adversary* $D = (D_1, D_2)$ against Π operates in three stages. First, D_1 gets as input a bit b and outputs a vector of messages $\mathbf{x}$. Then, D_2 gets as input the public key pk, a vector of ciphertexts $\mathbf{c}$, and ouptuts a "guess" bit d. To an encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ and a DH-CPA adversary $D = (D_1, D_2)$ we associate

```
**Experiment** $\mathbf{Exp}^{\text{dh-cpa}}_{\Pi,A}(k)$:
    $b \xleftarrow{\$} \{0,1\}$ ; $(\mathbf{x}, t) \xleftarrow{\$} D_1(b)$
    $\mathbf{c} \xleftarrow{\$} \mathcal{E}(pk, \mathbf{x})$
    $d \xleftarrow{\$} D_2(pk, \mathbf{c})$
    If $b = d$ return 1 else return 0
```

As before, we require D_1's output to satisfy $|\mathbf{x}_0| = |\mathbf{x}_1|$ and $|\mathbf{x}_0[i]| = \mathbf{x}_1[i]$ for all $1 \le i \le |\mathbf{x}_0|$. Moreover, we require that $\mathbf{x}_0[i_1] = \mathbf{x}_0[i_2]$ if and only if $\mathbf{x}_1[i_1] = \mathbf{x}_1[i_2]$ for all $1 \le i_1, i_2 \le |\mathbf{x}_0|$. Define the *DH-CPA advantage* of D against Π as

$$\mathbf{Adv}^{\text{dh-cpa}}_{\Pi,D}(k) = 2 \cdot \Pr\left[\,\mathbf{Exp}^{\text{dh-cpa}}_{\Pi,D}(k) \Rightarrow 1\,\right] - 1\ .$$

The following theorem establishes equivalence between DET-CPA and DH-CPA. (For simplicity, the theorem only shows that DH-CPA implies DET-CPA, which is the "interesting" directions; the other direction is straightforward.) Note that [38] show a similar equivalence but their reductions are not necessarily polynomial time, which is crucial in our context.

Theorem 4.1.1. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and let $A = (A_1, A_2)$ be a DET-CPA adversary with min-entropy μ. Then there is a DH-CPA adversary $D = (D_1, D_2)$ with min-entropy μ such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) \;\; \le \;\; 36 \cdot \mathbf{Adv}^{\text{dh-cpa}}_{\Pi,D}(k) + \left(\frac{3}{4}\right)^{-k}\ .$$

Furthermore, D has min-entropy $\mu - 4$ the running-time of D is the time for at most that for k executions of A (but 4 in expectation).

The high-level intuition for the proof is as follows. We first show that it suffices to consider DET-CPA adversaries for which A_2 outputs $(\mathbf{x}, t)$ where t is *boolean.* Now, we would like to use the fact if t is easy to guess from the encryption of $\mathbf{x}$ then, the encryption of $\mathbf{x}$ conditioned on (1) the output $(\mathbf{x}, t)$ of A_2 being such that $t = 1$ or (2) the output $(\mathbf{x}, t)$ of A_2 being such that $t = 0$ are easy to distinguish. However, one of these distributions may be hard to sample from and have low entropy. Therefore, we next show it suffices to consider DET-CPA adversaries for which t is not just boolean but also *balanced*, meaning the probability it is 0 or 1 is about the same. Then, we can easily sample from the above-mentioned distributions by repeatedly running A.

REDUCTION TO THE BOOLEAN CASE. Call a DET-CPA adversary A *boolean* if it outputs test strings of length 1. We first show that is suffices to consider boolean DET-CPA adversaries.

Proposition 4.1.2. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and $A = (A_1, A_2)$ be a DET-CPA adversary that outputs test strings of length ℓ. Then there is a boolean DET-CPA adversary $B = (B_0, B_1, B_2)$ such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) \;\; \leq \;\; 2 \cdot \mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) \; .$$

Furthermore, the running-time of B is the time to run A plus $O(\ell)$.

Proof. The proof is identical to an argument in [33] for the information-theoretic setting. Adversary B works as follows:

Alg $B_1(1^k)$:	**Alg** $B_2(r)$:	**Alg** $B_3(pk, \mathbf{c}, r)$:
$r \xleftarrow{\$} \{0,1\}^n$	$(\mathbf{x}, t) \xleftarrow{\$} A_1(1^k)$	$g \xleftarrow{\$} A_3(pk, \mathbf{c})$
Return r	Return $(\mathbf{x}, \langle t, r \rangle)$	Return $\langle g, r \rangle$

For $d \in \{0,1\}$, let A_d denote the event $\mathbf{Exp}^{\text{det-cpa-d}}_{\Pi,A}(k) \Rightarrow 1$ and similarly B_d denote $\mathbf{Exp}^{\text{det-cpa-d}}_{\Pi,B}(k) \Rightarrow 1$. Then

$$\begin{aligned}
\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) &= \Pr[\, B_1 \,] - \Pr[\, B_0 \,] \\
&= \Big(\Pr[\, A_1 \,] + \frac{1}{2} \cdot (1 - \Pr[\, A_1 \,]) \Big) \\
&\quad - \Big(\Pr[\, A_0 \,] + \frac{1}{2} \cdot (1 - \Pr[\, A_0 \,]) \Big) \\
&= \frac{1}{2} \cdot (\Pr[\, A_1 \,] - \Pr[\, A_0 \,]) \\
&= \frac{1}{2} \cdot \mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) \; .
\end{aligned}$$

The claimed running-time of B is easy to verify. □

REDUCTION TO THE BALANCED BOOLEAN CASE. The next step is to show that it in fact suffices to consider boolean DET-CPA adersaries that are *balanced*, meaning the probability the partial information is 1 or 0 is approximately 1/2. Namely, call a boolean DET-CPA adversary $A = (A_0, A_1, A_2)$ δ-*balanced* if for all $b \in \{0, 1\}$

$$\left| \Pr\left[\, t = b \;:\; (\mathbf{x}, t) \xleftarrow{\$} A_1(1^k, \mathit{state}) \,\right] - \frac{1}{2} \right| \leq \delta$$

for all *state* output by A_0.

Proposition 4.1.3. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and $B = (B_1, B_2)$ be a boolean DET-CPA adversary. Then for any $0 \leq \delta < 1/2$ there is a δ-balanced boolean DET-CPA adversary $B' = (B'_1, B'_2)$ such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) \;\; \leq \;\; \left(\frac{2}{\delta} + 1\right) \cdot \mathbf{Adv}^{\text{det-cpa}}_{\Pi,B'}(k) \; .$$

Furthermore, the running-time of B' is the time to run B plus $O(1/\delta)$.

Proof. We give a simplified proof due to [32], where for simplicity we assume $1/\delta$ is an integer. Adversary B' works as follows:

```
Algorithm $B_1(1^k)$:
$(\mathbf{x}, t) \xleftarrow{\$} A_1(1^k)$
$i \xleftarrow{\$} [2(1/\delta) + 1]$
If $i \leq 1/\delta$ then return $(\mathbf{x}, 0)$
Else if $i \leq 2(1/\delta)$ then return $(\mathbf{x}, 1)$
Else return $(\mathbf{x}, t)$
```

```
Algorithm $B_2(pk, \mathbf{c})$:
$g \xleftarrow{\$} A_2(pk, \mathbf{c})$
$j \xleftarrow{\$} [2(1/\delta) + 1]$
If $j \leq \delta$ then return 0
Else if $j \leq 2(1/\delta)$ then return 1
Else return $g$
```

Note that B is δ-balanced, since for all $b \in \{0, 1\}$

$$\left| \Pr\left[\, t = b \;:\; (\mathbf{x}, t) \xleftarrow{\$} A_1(1^k) \,\right] - \frac{1}{2} \right| \;\leq\; \frac{1}{2(1/\delta) + 1} \; .$$

As before, for $d \in \{0,1\}$, let A_d denote the event $\mathbf{Exp}^{\text{det-cpa-d}}_{\Pi,A}(k) \Rightarrow 1$ and similarly B_d denote $\mathbf{Exp}^{\text{det-cpa-d}}_{\Pi,B}(k) \Rightarrow 1$. Then

$$\begin{aligned}
\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) &= \Pr[\,B_1\,] - \Pr[\,B_0\,] \\
&= \Pr[\,B_1 \mid E\,] - \Pr[\,B_0 \mid E\,] + \Pr\left[\,B_1 \mid \overline{E}\,\right] - \Pr\left[\,B_0 \mid \overline{E}\,\right] \\
&= \Pr[\,B_1 \mid E\,] - \Pr[\,B_0 \mid E\,] + \frac{1}{2} - \frac{1}{2} \\
&= \frac{1}{2} \cdot \mathbf{Adv}^{\text{det-cpa}}_{\Pi,A}(k) \ .
\end{aligned}$$

As before, the claimed running-time of B' is easy to verify. □

Reduction to distribution hiding. The final component for the proof is as follows.

Proposition 4.1.4. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and $B = (B_1, B_2)$ be a δ-balanced boolean DET-CPA adversary. Then there is a DH-CPA adversary $D = (D_1, D_2)$ with min-entropy $\mu - \log(1 - 2\delta) + 1$ such that

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) \quad \leq \quad 2 \cdot \mathbf{Adv}^{\text{dh-cpa}}_{\Pi,D}(k) + \left(\frac{3}{4}\right)^{-k} \ .$$

Furthermore, the running-time of B is the time for at most k executions of A (but 4 in expectation).

Proof. Adversary D works as follows.

```
Algorithm $D_1(b)$:
For $i = 1$ to $k$ do:
    $(\mathbf{x}, t) \xleftarrow{\$} A_1(1^k)$
    If $t = b$ then return $\mathbf{x}$
Return $\mathbf{x}$

Algorithm $D_2(pk, \mathbf{c})$:
$g \xleftarrow{\$} A_2(pk, \mathbf{c})$
Return $g$
```

Let BAD denote the event that the final return statement is executed. Let $\mathsf{CORRECT}_D$ be the event that $b = d$ when D is executed in the DH-CPA experiment with Π and simiarly let $\mathsf{CORRECT}_B$ denote the event that $t = g$ when B is executed in the DET-CPA experiment with Π. Then

$$\begin{aligned}
\mathbf{Adv}^{\text{det-cpa}}_{\Pi,B}(k) &= \Pr\left[\,\mathsf{CORRECT}_B \mid t = 1\,\right] + \Pr\left[\,\mathsf{CORRECT}_B \mid t = 0\,\right] \\
&\leq \Pr\left[\,\mathsf{CORRECT}_B \mid t = 1 \;\wedge\; \overline{\mathsf{BAD}}\,\right] \\
&\quad + \Pr\left[\,\mathsf{CORRECT}_B \mid t = 0 \;\wedge\; \overline{\mathsf{BAD}}\,\right] + \Pr\left[\,\overline{\mathsf{BAD}}\,\right] \\
&= \Pr\left[\,\mathsf{CORRECT}_D \mid b = 1\,\right] \\
&\quad + \Pr\left[\,\mathsf{CORRECT}_D \mid b = 0\,\right] + \Pr\left[\,\overline{\mathsf{BAD}}\,\right] \\
&= \mathbf{Adv}^{\text{dh-cpa}}_{\Pi,D}(k) + \Pr\left[\,\overline{\mathsf{BAD}}\,\right] \\
&\leq \Pr\left[\,\mathsf{CORRECT}_D \mid b = 1\,\right] \\
&\quad + \Pr\left[\,\mathsf{CORRECT}_D \mid b = 0\,\right] + \left(\frac{1}{2} + \delta\right)^{-k}
\end{aligned}$$

where the last line uses that B is δ-balanced. So it remains to argue the min-entropy of D. Let $b \in \{0,1\}$, $i \in [v]$, and $x \in \{0,1\}^*$. Denote $\Pr\left[\,t = b \;:\; (\mathbf{x}, t) \xleftarrow{\$} A_1(1^k)\,\right]$ by $P_A(b)$, $\Pr\left[\,\mathbf{x}[i] = x \;:\; (\mathbf{x}, t) \xleftarrow{\$} A_1(1^k)\,\right]$ by $P_A(x,i)$, and
$\Pr\left[\,\mathbf{x}[i] = x \;\wedge\; t = b \;:\; (\mathbf{x}, t) \xleftarrow{\$} A_1(1^k)\,\right]$ by $P_A(x,i,b)$. We have

$$\begin{aligned}
\Pr\left[\,\mathbf{x}[i] = x \;:\; (\mathbf{x}, t) \xleftarrow{\$} D(b)\,\right] &= \left(\sum_{i=1}^{k-1} P_A(\overline{b})^{i-1} P_A(x,i,b)\right) + P_A(\overline{b})^{k-1} P_A(x,i,\overline{b}) \\
&= P_A(x,i,b)\frac{1 - P_A(\overline{b})^k}{P_A(b)} + P_A(\overline{b})^{k-1} P_A(x,i,\overline{b}) \\
&\leq \frac{1}{P_A(b)} \cdot \left(P_A(x,i,b) + P_A(x,i,\overline{b})\right) \\
&= \frac{1}{P_A(b)} \cdot P_A(x,i) \\
&= \frac{1}{1/2 - \delta} \cdot 2^{-\mu}
\end{aligned}$$

and the claim follows. □

Theorem 4.1.1 now follows by combining Propositions 4.1.2, 4.1.3, and 4.1.4 with $\delta = 1/4$.

4.2 Constructions in the Standard Model

The constructions presented in Section 3.3 are quite practical and achieve all the security we could hope for from a deterministic scheme. However, due to the well-known limitations of the random oracle model [23], we would like to study constructions that do not use random oracles. In this section, we relate the contruction of standard-model deterministic encryption to that of one-way trapdoor functions with a special kind of hardcore function we call *robust*. In particular, this leads to schemes achieving:

- Single-message (or block-source [41]) DET-CPA security from exponentially-hard trapdoor permutations.
- Single-message (or block-source [41]) DET-CPA security for uniform messages from exponentially-hard trapdoor functions.
- q-message DET-CPA security, where the public key is allowed to depend on q, based on lossy trapdoor functions. (More generally, the scheme achieves security for q-block-source, where every q messages we get "fresh entropy").

4.2.1 Robust Hardcore Functions

Extensions to one-wayness and hardcore functions. For our results, we extend the usual notions of one-wayness and hardcore functions to vectors of inputs drawn from non-unform distributions, similar to the case of deterministic encryption. To a trapdoor function generator $\mathcal{F}$ and inverter $I = (I_1, I_2)$ we associate

```
Experiment $\mathbf{Exp}^{\mathrm{owf}}_{\mathcal{F},I}(k)$:
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}$
$\mathbf{x} \xleftarrow{\$} I_1(1^k)$
$x' \xleftarrow{\$} I(f, f(\mathbf{x}))$
If $\exists i$ such that $\mathbf{x}[i] = x'$ return 1 else return 0
```

Define the *OWF advantage* of I against F as

$$\mathbf{Adv}^{\text{owf}}_{\mathcal{F},I}(k) = \Pr\left[\,\mathbf{Exp}^{\text{owf}}_{\mathcal{F},A}(k) \Rightarrow 1\,\right]\ .$$

We extend hardore functions in a similar way. Namely, to a trapdoor function generator $\mathcal{F}$, function hc, and distinguisher $D = (D_1, D_2)$ we associate

```
**Experiment** $\mathbf{Exp}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},D}(k)$:
$b \xleftarrow{\$} \{0,1\}$ ; $(f, f^{-1}) \xleftarrow{\$} \mathcal{F}$
$\mathbf{x} \xleftarrow{\$} D_1(1^k)$
$\mathbf{h}_0 \leftarrow \mathsf{hc}(f, \mathbf{x})$ ; $\mathbf{h}_1 \xleftarrow{\$} (\{0,1\}^n)^{\times|\mathbf{x}|}$
$d \xleftarrow{\$} D_2(f, f(\mathbf{x}), \mathbf{h}_b)$
If $d = b$ return 1 else return 0
```

For all $k \in \mathbb{N}$, define the *HCF advantage* of A against F, hc as

$$\mathbf{Adv}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},A}(k) = 2 \cdot \Pr\left[\,\mathbf{Exp}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},A}(k) \Rightarrow 1\,\right] - 1\ .$$

Analogously to the case of deterministic encryption, we say that an inverter $I = (I_1, I_2)$ that outputs vectors of length v has *min-entropy* μ if $\mathrm{H}_\infty(X_i) \geq \mu$ for all $1 \leq i \leq v$, where X_i is the random variable with the distribution of $\mathbf{x}[i]$ over $\mathbf{x} \xleftarrow{\$} I_1(1^k)$ and similarly for a distinguisher $D = (D_1, D_2)$.

Robustness. We define a new notion of *robust* hardcore functions. Intuitively, robust hardcore functions are those that remain one-way when the min-entropy of the input is slightly reduced. Since our reduction are concrete it is not necessary for us to define this notion formally. However, for completeness, the following is a possible quantitative definition.

Definition 4.2.1. Let $\mathcal{F}$ be a trapdoor function generator and be a hardcore function for entropy μ. We say that hc is *c-robust for entropy* μ if there for every HCF adversary A with entropy $\mu - c$ there is an HCF adversary B with entropy μ such that for every

$k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F},\mathsf{hc},A}(k) \;\; \leq \;\; 2^{O(c)} \cdot \mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F},B}(k) \; . \tag{21}$$

Furthermore, the running-time of B depends polynomially on the running-time of A.

It is illustrative to consider just for single-input distinguishers (i.e., for $|\mathbf{x}| = 1$ in the HCF experiment). Note here for example that every bit of the input to RSA is well-known to be hardcore assuming RSA is one-way [3]. However, they are not even 1-robust, since when we decrease the min-entropy of the input the bit may become fixed. Indeed, no hardcore function that depends only on the input and not on the description of the function itself can be hardcore by a similar argument.

4.2.2 The Encrypt-with-Hardcore Scheme

To gain some intuition, let us recall the Encrypt-with-Hash scheme in Section 3.3.1. The idea there is that in the random oracle model the hash of the message looks like independent random coins. Can we use a similar construction in the standard model? This motivates the following construction based on hardcore functions.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme, $\mathcal{F}$ be a trapdoor function generator, and hc be a hardcore function. Assume (e.g., by suitable padding) that hc has the property $\mathsf{hc}_f(x) \in \mathsf{Coins}_{pk}(|x|)$ for all pk output by $\mathcal{K}$ and all $x \in \{0,1\}^*$. Define the associated "*Encrypt-with-Hardcore*" deterministic encryption scheme
$\mathsf{EwHCore}[\Pi, \mathcal{F}, \mathsf{hc}] = (\mathcal{K}, \mathcal{E}, \mathcal{DD})$ with plaintext-space $\mathrm{PtSp} = \{0,1\}^k$ via

```
**Alg** $\mathcal{K}(1^k)$:
$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
$(f, f^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
Return $((pk, f), (sk, f^{-1})$
```

```
**Alg** $\mathcal{DE}((pk, f), x)$
$r \leftarrow \mathsf{hc}(f, x)$
$c \leftarrow \mathcal{E}(pk, f(x); r)$
Return $c$
```

```
**Alg** $\mathcal{DD}((sk, f^{-1}), c)$:
$y \leftarrow \mathcal{D}(sk, c)$
$x \leftarrow f^{-1}(y)$
Return $x$
```

Security analysis. Suppose hc is hardcore for $\mathcal{F}$ against adversaries with min-entropy μ. One might think that DET-CPA security of $\mathsf{EwHCore}[\Pi, \mathcal{F}, \mathsf{hc}]$ against adversaries with min-entropy μ then follows by by IND-CPA security of Π. However, this is not true. To see this, suppose hc is a physical hardcore function. Define $\Pi' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ to be like $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ except that the coins consumed by $\mathcal{E}'$ are extended by one bit, which $\mathcal{E}'$ outputs in the clear and $\mathcal{D}'$ ignores. That is, define $\mathcal{E}'(pk, x; r\|b) = \mathcal{E}(pk, x; r)\|b$ and $\mathcal{D}'(sk, y\|b) = \mathcal{D}(sk, y)$. Then IND-CPA security of Π' follows from that of Π, but a straightforward attack shows $\mathsf{EwHCore}[\Pi, \mathcal{F}, \mathsf{hc}]$ is not DET-CPA. This is where our notion of robustness comes into play.

Theorem 4.2.2. Let $D = (D_1, D_2)$ be a single-message DH-CPA adversary against $\mathsf{EwHash}[\Pi]$ with min-entropy μ. Then there is an IND-CPA adversary A against Π, and HCF adversaries $B = (B_1, B_2)$ against $\mathcal{F}, \mathsf{hc}$ with min-entropy μ, such that

$$\mathbf{Adv}^{\text{dh-cpa}}_{\mathsf{EwHCore},D}(k) \quad \leq \quad \mathbf{Adv}^{\text{ind-cpa}}_{\Pi,A}(k) + 2 \cdot \mathbf{Adv}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},B}(k) \; . \tag{22}$$

Furthermore, the running-times of A, B are the time to run D.

Proof. Let Game G_1 correspond to the DH-CPA experiment with D against $\mathsf{EwHCore}$, and let Game G_2 be like G_1 except that the coins used to encrypt the challenge plaintext vector are truly random. For $i \in \{0, 1\}$ let $B^i = (B^i_1, B^i_2)$ be the HCF adversary against $\mathcal{F}$ hc defined via

```
**Algorithm** $B^i_1(1^k)$:       | **Alg** $B^i_2(pk, \mathbf{y}, \mathbf{h})$:
$\mathbf{x} \xleftarrow{\$} D_1(i)$ | $\mathbf{c} \leftarrow \mathcal{E}(pk, \mathbf{y}; \mathbf{h})$
Return $\mathbf{x}$                 | $d \xleftarrow{\$} D_2(pk, \mathbf{c})$
                                    | Return $d$
```

Then

$$\begin{aligned}
\Pr\left[\, G_1^D \Rightarrow b \,\right] \quad &= \quad \Pr\left[\, G_1^D \Rightarrow b \;\mid\; b = 1 \,\right] + \Pr\left[\, G_1^D \Rightarrow b \;\mid\; b = 0 \,\right] \\
&= \quad \Pr\left[\, G_2^D \Rightarrow b \;\mid\; b = 1 \,\right] + \mathbf{Adv}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},B^1}(k) \\
&\quad\quad + \Pr\left[\, G_2^D \Rightarrow b \;\mid\; b = 0 \,\right] + \mathbf{Adv}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},B^0}(k) \\
&\leq \quad \Pr\left[\, G_2^D \Rightarrow b \,\right] + 2 \cdot \mathbf{Adv}^{\text{hcf}}_{\mathcal{F},\mathsf{hc},B}(k)
\end{aligned}$$

where we take B to be whichever of B^0, B^1 has the larger advantage. Now define IND-CPA adversary A against Π via

```
Algorithm $A_1(pk)$:              | Alg $A_2(pk, \mathbf{c})$:
$\mathbf{x}_0 \xleftarrow{\$} D_1(0)$  | $d \xleftarrow{\$} D_2(pk, \mathbf{c})$
$\mathbf{x}_1 \xleftarrow{\$} D_1(1)$  | Return $d$
Return $(\mathbf{x}_0, \mathbf{x}_1)$ |
```

Then Equation 22 follows from Equation 22 from taking into account the definition of the advantages of D, A. □

By Theorem 4.1.1, we can now conclude DET-CPA security of EwHCore for plaintexts of 2 bits greater entropy, meaning 2-robustness of the hardcore function suffices for security of EwHCore. Using an alternative reduction from DET-CPA to DH-CPA by Fehr [41] it is possible to improve this to 1-robustness. We have thus reduced the search for secure deterministic encryption schemes in the standard model to hardcore functions that are robust in the above sense.

A subtle point worth mentioning is where we have used the fact that we use DH-CPA instead of DET-CPA security in the above proof. It is in the step that uses security of the hardcore function. If we used DET-CPA security, in this step the constructed HCF adversaries against $\mathcal{F}$ would need to test whether the output of the DET-CPA adversary against EwHCore is equal to a "target value" representing partial information on the input to $\mathcal{F}$, which these adversaries are not given.

4.3 Instantiations

Here we provide several instantiations of robust hardcore functions and hence of the Encrypt-with-Hardcore scheme. These instantiations are given for single inputs, i.e., $|\mathbf{x}| = 1$ in the OWF and HCF experiments, and are based on the observations that the Golreich-Levin hardcore function [46] is robust for any one-way function and a universal hash function is robust for lossy trapdoor functions [68]. These robust hardcore functions provide instantiations of the Encrypt-with-Hardcore scheme secure for encrypting a single message. By the results of [41] security for multiple messages distributed according to a block-source follows.

4.3.1 Instantiations based on Exponential One-Wayness

Here we present instantiations from computational hardness of inversion. The schemes obtained are for the single-message case (or equivalently block-sources [41]) only. The results rely on the following lemma.

Lemma 4.3.1. [40, Lemma 4] Let X be a random variable on a set S such that $\mathrm{H}_\infty(X) \geq \log|S| - \ell$. Then there is an event E such that $\Pr[\,E\,] \geq 2^{-\ell}$ and $U \mid E$ has the same distribution as X, where U is uniform and independent on S.

By combining Lemma 4.3.1 with the Goldreich-Levin Theorem 2.3.1, we obtain robust hardcore functions for one-way permutations and one-way functions for distributions with sufficient min-entropy relative to the hardness of inversion for inversion to still remain infeasible. Indeed, it furthermore follows that the Goldreich-Levin hardcore function is c-robust for such min-entropy for small c. Details follow.

Instantiations from trapdoor permutations. In the case of trapdoor permutations, we can use Blum-Micali-Yao [15, 78] iteration to extract many hardcore bits. Namely, let $\mathcal{F}$ be a trapdoor permutation generator. For $i \in \mathbb{N}$ denote by $\mathcal{F}^i$ the trapdoor permutation generator that iterates $\mathcal{F}$ i times, i.e., $(f^i, f^{-i}) \overset{\$}{\leftarrow} \mathcal{F}^i$. For f output by $\mathcal{F}$ define the Blum-Micali-Yao [15, 78], Goldreich-Levin [46] function $\mathcal{BMY}^i\colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^i$ via

$$\mathcal{BMY}^i(r,x) = \langle x, r\rangle \| \langle f(x), r\rangle \| \ldots \| \langle f^{i-1(x)}, r\rangle$$

By combining Theorem 2.3.1 and Theorem 4.3.1 (and noting that min-entropy is preserved under permutation), it follows that if $\mathcal{F}$ is sufficiently hard to invert that it remains one-way on distributions of entropy μ, then $\mathcal{BMY}^i$ is a c-robust hardcore function of $\mathcal{F}$ for μ and any $c = O(\log k)$. More generally, we have the following result.

Theorem 4.3.2. Let $\mathcal{F}$ be a trapdoor permutation generator. For any $i \in N$, let D be a single-input distinguisher against $\mathcal{BMY}^i$ with min-entropy $\mu = k - \ell$. Then there is single-input inverter I with min-entropy $\mu = k - \ell$ such that

$$\mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F}[\mathcal{GL}],\mathcal{BMY}^i,D}(k) \quad \leq \quad i \cdot 2^{\ell+3} \cdot \mathbf{Adv}^{\mathrm{owf}}_{\mathcal{F},I}(k) \; .$$

Furthermore, the running-time of I is the time for $O(\varepsilon^{-4}k^3)$ executions of D where $\varepsilon = \mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F}[\mathcal{GL}],\mathcal{BMY}^i,D}(k)$.

In the proof, we condition on the event E given by 4.3.1, which tells us that the challenge input for I has the right distribution. By Theorem 4.1.1 and Theorem 4.2.2, the above implies an instantiation of the Encrypt-with-Hardcore scheme from trapdoor permutations. In the case of standard one-way trapdoor permutations, we the instantiation can only tolerate nearly uniform messages. However, if the trapdoor permutation is *exponentially hard*, in the sense that efficient adversaries have exponentially small (in k) advantage against it (so that the reduction Theorem 4.3.2 is still meaningful for large ℓ) then the instantiation can tolerate correspondingly high min-entropy messages as well.

Instantiation from very hard trapdoor functions. Whereas in the trapdoor permutation case we can obtain DET-CPA secure encryption of a single high-entropy message from exponential hardness, in the trapdoor function case in general we need exponential hardness even to obtain DET-CPA secure encryption of a single *uniform* message (however, any "hardness" beyond that can be used to reduce the input entropy required). Similar to the case of trapdoor permutations, this is implied by the following result.

Theorem 4.3.3. Let $\mathcal{F}$ be a trapdoor function generator. For $i \in \mathbb{N}$, let D be a single-input distinguisher against $\mathcal{F}[\mathcal{GL}^i], \mathcal{GL}^i$ with min-entropy $\mu = k - \ell$. Then there is an inverter I with min-entropy μ such that for every $k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F}[\mathcal{GL}^i],\mathcal{GL}^i,D}(k) \;\leq\; 2^{\ell+i+3} \cdot \mathbf{Adv}^{\mathrm{owf}}_{\mathcal{F},I}(k) \; .$$

Furthermore, the running-time of I is the time for $O(\varepsilon^{-4}k^3)$ executions of D where $\varepsilon = \mathbf{Adv}^{\mathrm{hcf}}_{\mathcal{F}[\mathcal{GL}^i],\mathcal{GL}^i,D}(k)$.

As before, the proof simply combines Theorem 2.3.1 and Theorem 4.3.1. By Theorem 4.1.1 and Theorem 4.2.2, the above implies an instantiation of the Encrypt-with-Hardcore scheme from exponentially-hard trapdoor functions. In general, to make sure we have enough hardcore bits for the "outer" encryption scheme, we can expand the length of the hardcore function by using its output as a seed for a pseudorandom generator. Note that pseudorandom generators can be constructed in theory based on any one-way function [50], so they do not constitute an extra complexity assumption.

4.3.2 Instantiation Based on Lossiness

Peikert and Waters [68] showed that lossy trapdoor functions admit a very simple hardcore function, namely a universal hash function. We observe that this hardcore function, like that of Goldreich and Levin, is in fact robust in our sense. That is, it is hardcore if the input distribution merely has high entropy. Indeed, the following result follows directly from the Generalized Leftover Hash Lemma (for average conditional min-entropy) in [36].

Theorem 4.3.4. Let $\mathsf{LTDF} = (\mathcal{F}, \mathcal{F}')$ be a lossy trapdoor function with residual leakage s. Let $H\colon \mathcal{K} \times \{0,1\}^k \to \{0,1\}^n$ be a universal hash function. For any $\varepsilon > 0$, let D be a distinguish with min-entropy $s + n + 2\log(1/\varepsilon)$. Then there is a distinguisher D' with the same min-entropy such that

$$\mathbf{Adv}^{\mathrm{hcf}}_{\mathsf{LTDF}[\mathcal{H}],\mathcal{H},A}(k) \quad\leq\quad \mathbf{Adv}^{\mathrm{ltdf}}_{\mathsf{LTDF},D}(k) + \varepsilon \; .$$

Furthermore, the running time of D' is the time to run D.

By Theorem 4.1.1 and Theorem 4.2.2, the above implies an instantiation of the Encrypt-with-Hardcore scheme from lossy TDFs, which can tolerate plaintexts with min-entropy slightly more than the residual lossiness of the TDF plus the number of coins consumed by the probabilistic encryption scheme. We also present improved constructions, both in terms of efficiency and allowed distributions on the plaintexts, of deterministic encryption from lossy trapdoor functions below.

4.4 Improved Schemes from Lossy Trapdoor Functions

It turns out that we can obtain better constructions from lossy trapdoor functions, both in terms of efficiency and allowed distributions on the plaintexts. These improvements are based on a variant of the Leftover Hash Lemma introduced by Dodis and Smith [37], called the "Crooked" Leftover Hash Lemma.

4.4.1 Crooked Leftover Hash Lemma and Extensions

We recall a variant of the Leftover Hash Lemma due to Dodis and Smith [37]. The idea is as follows. Suppose we apply a pairwise independent function that is not necessary compressing to a high-entropy source X. Since it is not compressing, the output is not necessarily close to uniform. However, it looks so when composed with any "shrinking" function.

Lemma 4.4.1. (Crooked Leftover Hash Lemma) [37] Let $H\colon \mathcal{K} \times D \to R$ be a pairwise independent function with range R, and let $f : R \to S$ be a function to a set S. Let X be a random variable over D. Then

$$\Delta((K, f(H(K,X))), (K, f(U)) \;\le\; \frac{1}{2}\sqrt{|S|} \cdot 2^{-\mathrm{H}_\infty(X)/2}$$

where $K \xleftarrow{\$} \mathcal{K}$ and U is uniform over R.

For our results we extend the lemma in two ways, as well as give a simpler proof as compared to [37]. First, we strengthen the lemma to the case of t-wise independent functions, inspired by [55] who give a similar strengthening to the standard leftover hash lemma to the case of 4-wise independence. Second, we consider a relaxation to *almost* t-wise independence, inspired by [35] who show a similar relaxation for the standard leftover hash lemma. Namely, say that H is *δ-almost q-wise independent* if for all distinct $x_1, \ldots, x_q \in D$

$$\Delta((H(K,x_1),\ldots,H(K,x_q)),(U_1,\ldots,U_q)) \;\le\; \delta\,.$$

Our extensions to the crooked LHL are captured in the following.

Lemma 4.4.2. (Extended Crooked Leftover Hash Lemma) Let $H\colon \mathcal{K}\times D \to R$ be a $2t$-wise independent function for $t > 0$ with range R, and let $f : R \to S$ be a function. Let $\mathbf{X} = (X_1, \ldots, X_t)$ where the X_i are random variables over D such that $\mathrm{H}_\infty(X_i) \ge \mu$ for all $1 \le i \le n$ and moreover $\Pr[\,X_i = X_j\,] = 0$ for all $1 \le i \ne j \le t$. Then

$$\Delta((K, f(H(K,\mathbf{X}))),(K, f(\mathbf{U}))) \;\le\; \frac{1}{2}\sqrt{|S|^t(t^2 2^{-\mu} + 3\delta)}$$

where $K \xleftarrow{\$} \mathcal{K}$ and $\mathbf{U} = (U_1, \ldots, U_t)$ where the U_i are all uniform and independent over R (recall that functions operate on vectors component-wise).

Proof. Writing $\mathbf{E}_k$ for the expectation over the choice of k according to the distribution of K, it follows that

$$\begin{aligned}
\Delta\big((K, f(H(K,\mathbf{X}))),(K, f(\mathbf{U}))\big) &= \mathbf{E}_k\big[\Delta\big(f(H(k,\mathbf{X})), f(\mathbf{U})\big)\big] \\
&\le \frac{1}{2}\mathbf{E}_k\left[\sqrt{|S|^t \cdot D\big(f(H(k,\mathbf{X})), f(\mathbf{U})\big)}\right] \\
&\le \frac{1}{2}\sqrt{|S|^t \cdot \mathbf{E}_k\big[D\big(f(H(k,\mathbf{X}))), f(\mathbf{U})\big)\big]}
\end{aligned}$$

where the second inequality is due to Jensen's inequality. We will show that

$$\mathbf{E}_k\big[D\big(f(H(k,\mathbf{X})), f(\mathbf{U})\big)\big] \le\; t^2 2^{-\mu} + 3\delta\,,$$

which completes the proof. Write $\mathbf{Y} = H(k, \mathbf{X})$ for an arbitrary but fixed k. Then

$$\begin{aligned} D\big(f(\mathbf{Y}), f(\mathbf{U})\big) &= \sum_{\mathbf{s}} \big(P_{f(\mathbf{Y})}(\mathbf{s}) - P_{f(\mathbf{U})}(\mathbf{s})\big)^2 \\ &= \sum_{\mathbf{s}} P_{f(\mathbf{Y})}(\mathbf{s})^2 - 2\sum_{\mathbf{s}} P_{f(\mathbf{Y})}(\mathbf{s}) P_{f(\mathbf{U})}(\mathbf{s}) + \mathrm{Col}(f(\mathbf{U})) \ . \end{aligned}$$

For a set $Z \subseteq R^t$, define $\delta_{\mathbf{r},Z}$ to be 1 if $\mathbf{r} \in Z$ and else 0. For $\mathbf{s} \in S^t$ we can write $P_{f(\mathbf{Y})}(\mathbf{s}) = \sum_{\mathbf{x}} P_{\mathbf{X}}(\mathbf{x}) \delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})}$ and thus

$$\begin{aligned} \sum_{\mathbf{s}} P_{f(\mathbf{Y})}(\mathbf{s})^2 &= \sum_{\mathbf{s}} \Big(\sum_{\mathbf{x}} P_{\mathbf{X}}(\mathbf{x}) \delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})}\Big)\Big(\sum_{\mathbf{x}'} P_{\mathbf{X}}(\mathbf{x}') \delta_{H(k,\mathbf{x}'),f^{-1}(\mathbf{s})}\Big) \\ &= \sum_{\mathbf{s},\mathbf{x},\mathbf{x}'} P_{\mathbf{X}}(\mathbf{x}) P_{\mathbf{X}}(\mathbf{x}') \delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})} \delta_{H(k,\mathbf{x}'),f^{-1}(\mathbf{s})} \ , \end{aligned}$$

so that

$$\begin{aligned} \mathbf{E}_k\Big[\sum_{\mathbf{s}} P_{f(\mathbf{Y})}(\mathbf{s})^2\Big] &= \sum_{\mathbf{s}} \sum_{\mathbf{x},\mathbf{x}'} P_{\mathbf{X}}(\mathbf{x}) P_{\mathbf{X}}(\mathbf{x}') \mathbf{E}_k[\delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})} \delta_{H(k,\mathbf{x}'),f^{-1}(\mathbf{s})}] \\ &= \sum_{\mathbf{s}} \sum_{\exists i,j,\, \mathbf{x}[i]=\mathbf{x}'[j]} P_{\mathbf{X}}(\mathbf{x}) P_{\mathbf{X}}(\mathbf{x}') \\ &\quad + \sum_{\mathbf{s}} \sum_{\forall i,j,\, \mathbf{x}[i]\neq\mathbf{x}'[j]} P_{\mathbf{X}}(\mathbf{x}) P_{\mathbf{X}}(\mathbf{x}') \mathbf{E}_k[\delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})} \delta_{H(k,\mathbf{x}'),f^{-1}(\mathbf{s})}] \\ &\leq t^2 2^{-\mu} + \mathrm{Col}(f(U)) + \delta \end{aligned}$$

where the first term is by a union bound over all $1 \leq i, j \leq t$ and for the remaining terms we use the δ-almost $2t$-wise independence of H and note that

$$E_k[\delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})} \delta_{H(k,\mathbf{x}'),f^{-1}(\mathbf{s})}] = \Pr\left[\, f(H(K,\mathbf{x})) = f(H(K,\mathbf{x}')) \,\right] \ .$$

Similarly,

$$\begin{aligned} \sum_{\mathbf{s}} P_{f(\mathbf{Y})}(\mathbf{s}) P_{f(\mathbf{U})}(\mathbf{s}) &= \sum_{\mathbf{s}} \Big(\sum_{\mathbf{x}} P_{\mathbf{X}}(\mathbf{x}) \delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})}\Big)\Big(\frac{1}{|R|} \sum_{\mathbf{u}} \delta_{\mathbf{u},f^{-1}(\mathbf{s})}\Big) \\ &= \frac{1}{|R|} \sum_{\mathbf{s}} \sum_{\mathbf{u},\mathbf{x}} P_{\mathbf{X}}(\mathbf{x}) \delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})} \delta_{\mathbf{u},f^{-1}(\mathbf{s})} \end{aligned}$$

so that

$$\begin{aligned} \mathbf{E}_k\Big[\sum_{\mathbf{s}} P_{f(\mathbf{Y})}(\mathbf{s}) P_{f(\mathbf{U})}(\mathbf{s})\Big] &= \frac{1}{|R|} \sum_{\mathbf{s}} \sum_{\mathbf{u},\mathbf{x}} P_{\mathbf{X}}(\mathbf{x})\, \mathbf{E}_k[\delta_{H(k,\mathbf{x}),f^{-1}(\mathbf{s})} \delta_{\mathbf{u},f^{-1}(\mathbf{s})}] \\ &\geq \mathrm{Col}(f(\mathbf{U})) - \delta \end{aligned}$$

using δ-almost t-wise independence of H. By combining the above, it follows that

$$\mathbf{E}_k\big[D\big(f(\mathbf{Y}), f(\mathbf{U})\big)\big] \;\leq\; t^2 2^{-\mu} + 3\delta$$

which was to be shown. $\square$

Remark 4.4.3. For our main construction, we will need to extend Lemma 4.4.2 to the case that H is a *permutation*. In this case, say that H is *δ-almost q-wise independent* if for all distinct $x_1, \ldots, x_q \in D$

$$\Delta((H(K, x_1), \ldots, H(K, x_q)), (P_1, \ldots, P_q)) \;\leq\; \delta$$

where P_1 is uniform and P_i for $i > 1$ is uniform on the set of points not in the outcomes of $P_1, \ldots, P_{i-1}$. (i.e., $P_1, \ldots, P_q$ are correlated.) It is straightforward to verify from the proof that in this case Lemma 4.4.2 holds when replacing $(U_1, \ldots, U_t)$ with $(P_1, \ldots, P_t)$ in the lemma statement as defined above.

4.4.2 The General Scheme

The paradigm for the construction goes back to the work of Dodis and Smith [38, 37]. Namely, they show that randomness extractors, in addition to producing a uniform output, also *hide partial information about the input.* However, it is not immediately clear how to apply this paradigm in the public-key context. To illustrate the basic idea, we first present a simpler scheme due to Fehr [41].

A WARM-UP SCHEME. Let us call a lossy trapdoor function $\mathsf{LTDF} = (\mathcal{F}, \mathcal{F}')$ *universal* if $\mathcal{F}'$ implements a universal function family, meaning the function $\mathcal{H}\colon \mathcal{F}' \times \{0,1\}^k \to R$ where R is the range of f' output by $\mathcal{F}'$ is universal. For example, it is not hard to see that the DDH-based LTDF of Peikert and Waters [68] has this property. The claim is then LTDF viewed as an encryption scheme is itself DET-CPA secure. That is, define the associated deterministic encryption scheme $\Pi[\mathsf{LTDF}] = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where $\mathcal{K}$ on input 1^k returns $(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$, $\mathcal{E}$ on inputs f, m returns $f(m)$ and $\mathcal{D}$ on inputs f^{-1}, c returns $f^{-1}(c)$. Then we have the following.

Theorem 4.4.4. [41] Let LTDF be a universal lossy TDF with residual leakage s. Then for any ε and any single-message DET-CPA adversary A against $\Pi[\mathsf{LTDF}]$ with min-entropy $\mu \geq s + 2\log(1/\varepsilon) + 2$ there is a distinguisher D against LTDF such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{det-cpa}}_{\Pi[\mathsf{LTDF}]}(k) \quad \leq \quad \mathbf{Adv}^{\text{ltdf}}_{\mathsf{LTDF},D}(k) + \varepsilon \ ,$$

Furthermore, the running-time of D is the time to run A.

The proof is a simple hybrid argument that concludes using Lemma 2.2.1. It is instructive to compare Theorem 4.4.4 with Theorem 4.3.4. Not only is the scheme here more efficient but the bound on the required entropy of the input is much better, as the construction tolerates plaintexts with entropy slightly more than the residual lossiness of the lossy TDF. The price to pay is that the Theorem 4.4.4 requires the extra condition that the lossy TDF be universal. In the general scheme below, we show how to drop this extra requirement.

The general scheme. Intuitively, in the general construction we handle q-bounded DET-CPA adversaries by modifying the warm-up scheme to first pre-process an input message using a $2q$-wise independent permutation, and appealing to Lemma 4.4.1 in the security proof. The catch is that for $q > 1$ such a permutation is not known to exist (in an explicit and efficiently computable sense). However, there are good constructions of *almost* $2q$-wise independent permutations. Namely, for any $t, \delta > 0$, Kaplan et al. [52] construct a t-wise δ-almost independent permutation whose key length is $O(tk + \log(1/\delta))$.

Let $\mathsf{LTDF} = (\mathcal{F}, \mathcal{F}')$ be a lossy trapdoor function and let $\mathcal{P}\colon \mathcal{K} \times \{0,1\}^k \to \{0,1\}^k$ be a family of permutations on k bits. Define the associated deterministic encryption scheme $\Pi[\mathsf{LTDF}, \mathcal{P}] = (\mathcal{K}, \mathcal{DE}, \mathcal{DD})$ with plaintext-space $\mathrm{PtSp} = \{0,1\}^k$ via

```
Alg $\mathcal{K}(1^k)$:                              | Alg $\mathcal{DE}((f,K),x)$   | Alg $\mathcal{DD}((sk,f^{-1}),c)$:
$(f,f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$ ; $K \xleftarrow{\$} \mathcal{K}$ | $c \leftarrow f(\mathcal{P}(K,x))$ | $x \leftarrow f^{-1}(\mathcal{P}^{-1}(K,c))$
Return $((f,K),(f^{-1},K))$                          | Return $c$                     | Return $x$
```

We have the following theorem.

Theorem 4.4.5. Suppose LTDF has residual leakage s, and let $q, \varepsilon > 0$. Set $\delta = 2 \cdot \varepsilon^2/(3 \cdot 2^{qs})$ and suppose $\mathcal{P}$ is δ-almost $2q$-wise independent. Then for any q-message DET-CPA adversary A with min-entropy $\mu \geq qs + 2\log q + 2\log(1/\varepsilon) - 1$, there is a LTDF distinguisher D such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{dh-cpa}}_{\Pi[\mathsf{LTDF},P],A}(k) \;\leq\; \mathbf{Adv}^{\text{ltdf}}_{\mathsf{LTDF},D}(k) + \varepsilon \,.$$

Furthermore, the running-time of D is the time to run A.

As before, the proof is a simple hybrid argument. In the case that $\mathcal{P}$ is perfectly pairwise independent (i.e., $q = 1$ and $\delta = 0$) we can conclude by Lemma 4.4.1. This gives a version of Theorem 4.4.4 that drops the extra universaily requirement on LTDF (without any change in parameters). In the general case we use Lemma 4.4.2 coupled with Remark 4.4.3. with $t = 2q$. Note that in this case the required entropy from the input is worse due to the factor q on s in the assumption. As a consequence, we need to use an LTDF with residual leakage $k \cdot o(1)$ residual leakage, namely less than k/q. Fortunately, there are such constructions under standard assumptions. In particular, the DDH-based LTDF of Peikert and Waters [68] satisfies this requirement. We also give a more efficient Paillier-based instantiation below.

Extension to q-block-sources. We note that the security proof for the construction can actually be extended to an unbounded number of plaintexts drawn from what we call a *q-block-source*, a generalization of a block-source where every q messages introduces some "fresh entropy." That is, we call $(X_1, \ldots, X_{qn})$ a q-block-source with entropy μ if for all $1 \leq i \leq n$, all $0 \leq j \leq q-1$, and all $x_1, \ldots, x_{qi-1}$ in the support of these random variables $\mathrm{H}_\infty(X_{qi+j} \mid X_1 = x_1, \ldots, X_{qi-1} = x_{qi-1})$. Security for such plaintext distributions follows from the fact that 4.4.2 extends to q-block-sources in the same way the original Leftover Hash Lemma extends to block sources [27].

Comparison with RSA-DOAEP. Additionally, we contrast our scheme here with the random oracle model RSA-DOAEP scheme in Subsection 3.3.2. Indeed, both schemes first pre-process an input plaintext before applying a trapdoor function, in the former case a t-wise (almost) independent permutation following by a lossy trapdoor function and in the latter case a 3-round Feistel network followed by RSA (or, more generally, a partial-domain one-way trapdoor function). In fact, Naor and Reingold [62] show that *four-round* Feistel network on k-bit input where the first and last rounds are pairwise independent and the middle rounds are t-wise independent, is t-wise independent for any $t < 2^{k/4-O(1)}$ and $\delta \leq t^2/2^{k/2}$. While, that error-bound is not good enough to instantiate the above construction (as for the original Leftover Hash Lemma [35] is very sensitive to introducing such a error term, in particular in our case the latter must be much less that 2^{qs} where $k = qs$ in the above bound), we can still in some sense view our use of the RO model for deterministic encryption as a way to achieve efficient unbounded independence.

Efficient Paillier-based LTDF. We provide an efficient Pailler-based lossy trapdoor function that we can use to instantiate the above construction, based on earlier work by Fehr [41]. Let $\mathcal{RSA}$ be the RSA key-generator, i.e, that outputs $(N, (p, q))$ where $N = pq$ and p, q are random $k/2$-bit primes. Let $s \geq 1$ be polynomial in k. Our construction is actually based on extension of Paillier's scheme [65] to the group $\mathbb{Z}_{N^{s+1}}$ due to Damgård and Jurik [29], with some modifications in the spirit of Damgård and Nielsen [30, 31]. Namely, define $\mathsf{LTDF}_{\text{paillier}} = (\mathcal{F}_{\text{paillier}}, \mathcal{F}'_{\text{paillier}})$ via

```
**Algorithm** $\mathcal{F}_{\text{paillier}}$:
$(N,(p,q)) \xleftarrow{\$} \mathcal{RSA}(1^k)$
$a \xleftarrow{\$} \mathbb{Z}_N^*$
$g \leftarrow (1+N)a^{N^s} \bmod N^{s+1}$
Return $((g,N),(p,q))$
```

```
**Alg** $\mathcal{F}'_{\text{paillier}}$:
$(N,(p,q)) \xleftarrow{\$} \mathcal{K}(1^k)$
$a \xleftarrow{\$} \mathbb{Z}_N^*$
$g \leftarrow a^{N^s} \bmod N^{s+1}$
Return $(g,N)$
```

where the evaluation of $(x,y) \in \mathbb{Z}_N^s \times \mathbb{Z}_N^*$ on input (g,N) is defined as $g^x y^{N^s} \bmod N^{s+1}$, and inversion of $y \in \mathbb{Z}_{N^{s+1}}$ on input (p,q) uses the decryption procedure of [29] to recover x, then uniquely recovers y as the N^s-th root of $c/g^x \bmod N$ (which can be computed efficiently given p,q) and returns (x,y).

Indistinguishability of the public keys in the lossy and injective modes follows from the Decisional Composite Residuosity Assumption of [65], as in [29]. Note that while in injective mode the range of the function is $\mathbb{Z}_{N^{s+1}}$, in lossy mode it consists of N-th powers so is isomorphic to $\mathbb{Z}_N^*$. Thus, the construction achieves lossiness sk or residual leakage k bits. Moreover, as compared to the constructions of [68], it has key-size $O(k)$ rather than $O(k^2)$.[1] We note that it is easy to extend the above construction to an all-but-one TDF as defined in [68], which will be useful to us in Chapter 5 where we address adaptivity or chosen-ciphertext security.

[1]Although Boyen and Waters [19] recently showed how to reduce the key-size for the DDH-based construction of [68] to $O(k)$ using bilinear maps.

CHAPTER V

ADAPTIVE TRAPDOOR FUNCTIONS

In this chapter, we consider a strengthening to one-wayness for trapdoor functions along a different dimension we call *adaptivity*. Intuitively, adaptivity means that the function remains one-way even in the presence of an inversion oracle that may be queried on some points in the range. (The terminology follows [67] who consider adaptivity for unkeyed one-way functions, without a trapdoor.) Our main result is that adaptivity actually serves to *weaken* the general assumptions on which we know how to build black-box chosen-ciphertext secure public-key encryption. It also unifies the recent schemes of [72, 68] based on stronger assumptions.

Another interpretation of our results is that the notions of [72, 68] imply not only chosen-ciphertext security but adaptive TDFs, which seems much stronger (for example, it is known that semantically secure encryption does not even imply one-way trapdoor functions, at least in a black-box way [44]). Thus, we may still be far off from finding the weakest general assumption to imply black-box chosen-ciphertext security.

5.1 Adaptive Trapdoor and Tag-based Trapdoor Functions

We introduce our notion of adaptivity for trapdoor functions as well as an extension called *tag-based* trapdoor functions.

ADAPTIVE ONE-WAYNESS. Let $\mathcal{F}$ be a trapdoor function generator. To $\mathcal{F}$ and an inverter I with access to an oracle we associate

```
**Experiment** $\mathbf{Exp}^{\text{aowf}}_{\mathcal{F},I}(k)$:
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}$
$x \xleftarrow{\$} \{0,1\}^k$
$x' \xleftarrow{\$} I^{f^{-1}(\cdot)}(f, f(x))$
If $x = x'$ return 1 else return 0
```

where we require that I does not query $f(x)$ to its oracle. Define the *AOWF advantage* of I against F as

$$\mathbf{Adv}^{\text{aowf}}_{\mathcal{F},I}(k) = \Pr\left[\, \mathbf{Exp}^{\text{aowf}}_{\mathcal{F},A}(k) \Rightarrow 1 \,\right] .$$

Note that chosen-ciphertext secure deterministic encryption as defined in Chapter 3 can be viewed as a strengthening of adaptive one-wayness.

TAG-BASED ADAPTIVE ONE-WAYNESS. A *tag-based* TDF with tag-space $\{0,1\}^t$ for $t = t(k)$ is an algorithm $\mathcal{F}_{\text{tag}}$ that on input 1^k outputs $(f_{\text{tag}}, f_{\text{tag}}^{-1})$ where for every $t \in \{0,1\}^t$, $f(t,\cdot)$ is a function on $\{0,1\}^k$ and $f^{-1}(t,\cdot)$ is its inverse. To $\mathcal{F}_{\text{tag}}$ and inverter $I = (I_1, I_2)$ (the latter with access to an oracle) we associate

```
**Experiment** $\mathbf{Exp}^{\text{tb-aowf}}_{\mathcal{F},I}(k)$:
$(f_{\text{tag}}, f_{\text{tag}}^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$
$t \xleftarrow{\$} I_1(1^k)$
$x \xleftarrow{\$} \{0,1\}^k$
$x' \xleftarrow{\$} I^{f_{\text{tag}}^{-1}(\cdot,\cdot)}(f_{\text{tag}}, f_{\text{tag}}(t,x))$
If $x = x'$ return 1 else return 0
```

where we require I_2 does not make any query $f_{\text{tag}}^{-1}(t,\cdot)$ to its oracle. Define the *TB-AOWF advantage* of I against F as

$$\mathbf{Adv}^{\text{tb-aowf}}_{\mathcal{F},I}(k) = \Pr\left[\, \mathbf{Exp}^{\text{tb-aowf}}_{\mathcal{F},A}(k) \Rightarrow 1 \,\right] .$$

Note that the 'challenge tag" t is independent of f_{tag} and hence it may also be called selective-tag security (similar to selective-ID security for IBE schemes [24]).

Stronger variants of this security notion can be obtained by allowing the adversary choose the challenge-tag t adaptively. We note that typically one requires the size of the tag-space to be super-polynomial. In fact, adaptive tag-based TDFs with polynomial-size tag-space can be constructed from any OW-TDF, but are not sufficient for our applications.

5.2 Chosen-Ciphertext Secure Encryption from Adaptivity

We show that adaptive trapdoor functions and tag-based trapdoor functions lead to efficient and black-box chosen-ciphertext secure (probabilistic) encryption.

5.2.1 Constructions from Adaptive TDFs

We first show how to construct a one-bit CCA-secure public-key encryption scheme from an adaptive TDF. By a recent result of Myers and Shelat [61], this implies a black-box construction of a many-bit scheme as well.

Let $\mathcal{F}$ be a TDF generator and $\mathsf{hc}(\cdot)$ be a hardcore bit. We construct PKE scheme $\Pi[\mathcal{F}] = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with message-space $\{0,1\}$ as follows. Algorithm $\mathcal{K}$ outputs $(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$ and $\mathcal{E}$ and $\mathcal{D}$ are defined via

```
Algorithm $\mathcal{E}(f, x)$:
For $i = 1$ up to $k$ do
    $x \xleftarrow{\$} \{0,1\}^k$ ; $h \leftarrow \mathsf{hc}(x)$
    If $h = b$ then return $f(x)\|0$
Return $b\|1$
```

```
Algorithm $\mathcal{D}(f^{-1}, y\|\mathsf{flag})$:
If $\mathsf{flag} = 1$ then return $y$
Else return $\mathsf{hc}(f^{-1}(y))$
```

It is clear that the above construction satisfies *correctness.* (Note that if the encryption algorithm happens to output the message in the clear it is still correctly decrypted, so this is a security, not a functionality, concern.) We now turn to security.

Theorem 5.2.1. Let $A = (A_1, A_2)$ be an IND-CCA adversary against $\Pi[\mathcal{F}]$. Then there is a HCB adversary against $\mathcal{F}, \mathsf{hc}$ such that Then there is a distinguisher D

such that

$$\mathbf{Adv}^{\text{ind-cca}}_{\Pi[\mathcal{F}],A}(k) \quad \leq \quad k \cdot \left(\mathbf{Adv}^{\text{hcb}}_{\mathcal{F},\mathsf{hc},D}(k) + (k-1)q/2^{(k-1)/2} + 2^{-k}\right) \; . \tag{23}$$

Furthermore, the running-time of D is the time to execute A plus $O(k)$.

Optimizations. We note that our construction here can be simplified and made much more efficient if the given adaptive TDF is a permutation or has linearly many simultaneous hardcore bits. Namely, in this case one can use the adaptive TDF as a key-encapsulation mechanism for an IND-CCA-secure symmetric encryption scheme. (A key-encapsulation mechanism generates a ciphertext that encrypts a random symmetric key for use in hybrid encryption.) Additionally, for some *specific* hardcore bits one may be able to sample uniformly from the set $\{x \in \{0,1\}^k \mid \mathsf{hc}(x) = b\}$ more efficiently than by repeated sampling of the uniform distribution on $\{0,1\}^k$. (Indeed, this is the case for the Goldreich-Levin bit [46].) This translates to a corresponding efficiency improvement for the scheme.

Proof. As the messages space of $\Pi[\mathcal{F}]$ is $\{0,1\}$, we assume without loss of generality that A_1 always outputs $(0,1,\varepsilon)$. (That is, it chooses messages 0 and 1, and its state is empty.) We give a *multi-sample* adversary D' against hc; one can then obtain D via a standard hybrid argument. Adversary D' is given in Figure 10 and the games for the proof are given in Figure 11. We claim the following sequence of inequalities:

$$\begin{aligned}
\Pr\left[\, G_1^{A_2} \Rightarrow b \,\right] \quad &\leq \quad \Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] + \Pr[G_2^{A_2} \text{ sets } \mathsf{bad}_1\,] && (24)\\
&\leq \quad \Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] + \frac{kq}{2^{k/2}} && (25)\\
&\leq \quad \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \mathbf{Adv}^{\text{hcb}}_{\mathcal{F},\mathsf{hc},D}(k) + \frac{kq}{2^{k/2}} && (26)\\
&= \quad \Pr\left[\, G_4^{A_2} \Rightarrow b \,\right] + \Pr[G_3^{A_2} \text{ sets } \mathsf{bad}_2\,] + \mathbf{Adv}^{\text{hcb}}_{\mathcal{F},\mathsf{hc},D}(k) \\
&\qquad + \frac{kq}{2^{k/2}} && (27)\\
&= \quad \Pr\left[\, G_4^{A_2} \Rightarrow b \,\right] + 2^{-k} + \mathbf{Adv}^{\text{hcb}}_{\mathcal{F},\mathsf{hc},D}(k) + \frac{kq}{2^{k/2}} && (28)\\
&= \quad \frac{1}{2} + \frac{1}{2^k} + \mathbf{Adv}^{\text{hcb}}_{\mathcal{F},\mathsf{hc},D}(k) + \frac{(k-1)q}{2^{(k-1)/2}} && (29)
\end{aligned}$$

from which Equation 23 follows by multiplying both sides by 2 and subtracting 1, taking into account the definition of the advantage of A. Equation 24 is by Lemma 2.4.1. To see Equation 25, first note that A does not make a decryption query such that $c_1 = y_{i^*}$ by definition. For all $1 \leq i \neq i^* \leq k$, let Y_i be the random variable taking the value of y_i in the execution of Game G_2. Then conditioned on the view of A, the average min-entropy $\tilde{\mathrm{H}}_\infty(Y_i) \geq k-1$ by [36, Lemma 2.2(a)]. Therefore, by lemma [36, Lemma 2.2(b)] $\mathrm{H}_\infty(Y_i) \geq (k-1)/2$ with probability at least $1 - 2^{-(k-1)/2}$. By conditioning on this event and taking a union bound overall $i \neq i^*$ we get Equation 25. To see Equation 26, note that

$$\begin{aligned}
\Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] &= \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \left(\Pr\left[\, G_2^{A_2} \Rightarrow b \,\right] - \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right]\right) \\
&= \Pr\left[\, G_3^{A_2} \Rightarrow b \,\right] + \mathbf{Adv}^{\mathrm{hcb}}_{\mathcal{F},\mathsf{hc},D}(k)
\end{aligned}$$

where the last equality is by the definition of the advantage of D. As before, Equation 27 is by Lemma 2.4.1. To see Equation 28, note that each execution of the for-loop in the Initialize prodecure of Game G_3 chooses h uniformly and independently at random. Finally, Equation 29 is because A gets no information about b in Game G_4. $\square$

5.2.2 Constructions from Adaptive Tag-based TDFs

A BASIC SCHEME. Our construction of CCA-secure PKE from a adaptive tag-based TDFs is much simpler. It additionally makes use of a strongly one-time unforgeable signature scheme (see e.g. [72] for the definition). For simplicity, we give the construction below for the case of 1-bit messages. It is easy to extend it to a many-bit scheme, essentially by concatenating many applications of the TB-adaptive TDF under independent inputs but the same tag.

Let $\mathcal{F}_{\mathrm{tag}}$ be a tag-based trapdoor function generator and let $\mathsf{hc}(\cdot)$ be a hardcore bit. Let $\Sigma = (\mathcal{K}_\Sigma, \mathcal{S}, \mathcal{V})$ be a signature scheme whose verification keys are contained

```
**Algorithm** $D'^{f^{-1}(\cdot)}(f,(y_1,h_1),\ldots,(y_k,h_k))$:
$b \xleftarrow{\$} \{0,1\}$
Find the least $1 \le i^* \le k$ such that $h_{i^*} = b$
If there is no such $i^*$ then $c^* \leftarrow b\|1$
Else $c^* \leftarrow y_{i^*}\|0$
Run $A_2$ on inputs $f, c^*$:
    **On decryption query** $c$:
    $c_1\|\mathsf{flag} \leftarrow c$
    If $\exists i$ such that$c_1 = y_i$ then return $\perp$
    If $\mathsf{flag} = 1$ then return $c_1$
    Else return $\mathsf{hc}(f^{-1}(c_1))$
Let $d$ be the output of $A_2$
If $b = d$ return 1 else return 0
```

Figure 10: Adversary D for the proof of Theorem 5.2.1.

in the tag-space of $\mathcal{F}_{\text{tag}}$. We construct PKE scheme $\Pi[\mathcal{F}_{\text{tag}}, \Sigma] = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with message-space $\{0,1\}$ as follows:

```
**Algorithm** $\mathcal{E}(f_{\text{tag}}, b)$:
$(sk, vk) \xleftarrow{\$} \mathcal{K}_\Sigma(1^k)$
$x \xleftarrow{\$} \{0,1\}^k$
$y \xleftarrow{\$} f_{\text{tag}}(vk, x)$ ; $d \leftarrow \mathsf{hc}(x) \oplus b$
$\sigma \xleftarrow{\$} \mathcal{S}(sk, y\|d)$
Return $y\|d\|\sigma\|vk$
```

```
**Algorithm** $\mathcal{D}(f_{\text{tag}}^{-1}, y\|d\|\sigma\|vk)$:
If $\mathcal{V}(vk, c, \sigma) = 0$ then return $\perp$
Else return $\mathsf{hc}(f_{\text{tag}}^{-1}(vk, y)) \oplus d$
```

We have the following theorem.

Theorem 5.2.2. Let A be an IND-CCA adversary against $\Pi[\mathcal{F}_{\text{tag}}, \Sigma]$. Then there is a forger F against Σ and an HCB distinguisher D against $\mathcal{F}, \mathsf{hc}$ and such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{ind-cca}}_{\Pi[\mathcal{F}],A}(k) \quad \le \quad \mathbf{Adv}^{\text{sots}}_{\Sigma,F}(k) + \mathbf{Adv}^{\text{hcb}}_{\mathcal{F},\mathsf{hc},D}(k) \; .$$

Furthermore, the running-times of F, D are the time to run A.

```
procedure Initialize: $G_1, G_2$
$b \xleftarrow{\$} \{0,1\}$
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$
For $i = 1$ to $k$ do:
    $x \xleftarrow{\$} \{0,1\}^k$
    $h \leftarrow \mathsf{hc}(x)$
    If $h = b$ then
        Return $f, f(x) \| 0$
Return $f, b \| 1$
```

```
procedure Initialize: $G_3, \boxed{G_4}$
$b \xleftarrow{\$} \{0,1\}$
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$
For $i = 1$ to $k$ do:
    $x \xleftarrow{\$} \{0,1\}^k$
    $h \xleftarrow{\$} \{0,1\}$
    If $h = b$ then
        Return $f, f(x) \| 0$
$\mathsf{bad}_2 \leftarrow \mathsf{true}$ ; $\boxed{\text{return } \perp}$
Return $f, b \| 1$
```

```
On decryption query $c$: $G_1, \boxed{G_2}$
$c \leftarrow y \| \mathsf{flag}$
If $\exists i$ such that $c = y_i$ then
    $\mathsf{bad}_1 \leftarrow \mathsf{true}$ ; $\boxed{\text{return } \perp}$
If $\mathsf{flag} = 1$ then return $y$
Else return $\mathsf{hc}(f^{-1}(y))$
```

```
procedure Finalize($d$): All Games
If $b = d$ then return 1
Else return 0
```

Figure 11: Games for the proof of Theorem 5.2.1.

An optimized scheme. As in the case of adaptive TDFs, our construction of CCA-secure public-key encryption from adaptive tag-based TDFs can also be made much more efficient if the given TB-adaptive TDF is a permutation (for every tag) or has linearly many simultaneous hardcore bits. The idea is to first construct a selective-tag weakly CCA-secure tag-PKE scheme in the sense of [53] by using the adaptive tag-based TDF as a key-encapsulation mechanism for a one-time CPA-secure symmetric encryption scheme. Then, as shown in [53], we can apply the transform of Boneh et al. [17] to obtain a CCA-secure public-key encryption scheme, which uses only symmetric-key primitives. For completness we outline the construction in full below.

Namely, the transform of [17] uses a *message authentication code* (MAC) and an *encapsulation scheme*. Roughly speaking, an encapsulation scheme captures the properties of a one-time commitment scheme on a randomly generated message (the hiding and binding properties). However, one only requires the binding property to hold for honestly generated commitments. For concreteness, we use a specific instantiation from [17] (with a slightly improved analysis leading to better parameters).

Let $\mathcal{G}$ be an adaptive tag-based TDF and let $\mathsf{hc}(\cdot)$ be a hardcore function. (We assume here for simplicity that its output length is sufficiently long.) Let $\mathsf{SE} = (\mathcal{K}_S, \mathcal{E}_S, \mathcal{D}_S)$ be a symmetric encryption scheme. Let $H_1 \colon \{0,1\}^{368} \to \{0,1\}^{80}$ be a target-collision resistant hash function [13], and let $H_2 \colon \mathcal{K} \times \{0,1\}^{368} \to \{0,1\}^{128}$ be a universal hash function. Let $\mathcal{MAC} = (\mathcal{M}, \mathcal{V})$ be a message authentication code with 128-bit keys. Then define scheme $\Pi_{\text{opt}} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ associated to the above via

```
Alg $\mathcal{K}(1^k)$:
    $K \xleftarrow{\$} \mathcal{K}$
    $(g, g^{-1}) \xleftarrow{\$} \mathcal{G}$
    Return $((g, K), g^{-1})$

Alg $\mathcal{E}(pk, m)$:
    $x_1 \xleftarrow{\$} \{0,1\}^{368}$ ; $x_2 \xleftarrow{\$} \{0,1\}^k$
    $c_1 \leftarrow g(H(x_1), x_2)$
    $c_2 \leftarrow \mathcal{E}_S(\mathsf{hc}(x_2), m \| x_1)$
    $c_3 \leftarrow \mathcal{M}(H_2(x_1), c_1 \| c_2)$
    Return $H(x_1) \| c_1 \| c_2 \| c_3$

Alg $\mathcal{D}(sk, h \| c_1 \| c_2 \| c_3)$:
    $x_2 \leftarrow g^{-1}(h, c_1)$
    $m \| x_1 \leftarrow \mathcal{D}_S(\mathsf{hc}(x_2), c_2)$
    If $\mathcal{V}(H_2(x_1), c_1 \| c_2), c_3) = 1$
        Return $m$
    Else return $\perp$
```

In the construction, it is not hard to see that the c_1 and c_2 components of the ciphertext correspond to that of a selective-tag weakly CCA-secure encryption scheme in the sense of [53]. The proof of security then follows directly from [17], except that we improve the parameters here by using the Generalized Leftover Hash Lemma of [36]. Namely, we only need a hash function whose inputs are $368 = 160 + 80 + 128$ bits rather than 448 as in [17].

Witness-recoverability. We note that the optimized construction above is fully "witness-recovering" as defined in [68]; that is, via the decryption process the receiver recovers *all* of the randomness used by the sender to encrypt. (The constructs of [68] technically do not achieve this since, as the authors note, in their constructs the receiver does not recover the coins used to generate one-time signature keys.) In particular, it consistutes the first such scheme we are aware of without random oracles.

5.3 *Adaptivity from Stronger Trapdoor Functions*

Inspired by the constructions of CCA-secure encryption in [68, 72], we show that both adaptive TDFs and adaptive tag-based TDFs can be constructed in a black-box manner from correlated-product TDFs as introduced by Rosen and Segev [72]. As shown in [72, 60], lossy TDFs [68] imply correlated-product TDFs, thus by our result above they imply adaptive TDFs and adaptive tag-based TDFs too. However, we go on to show a much more efficient direct construction from lossy TDFs in combination with an all-but-one TDF as defined by [68].

5.3.1 Constructions from Correlated-Product TDFs

One-wayness under correlated-product. We recall the notion of one-wayness under correlated product [72]. Let $\mathcal{F}$ be a trapdoor function. To a trapdoor function generator $\mathcal{F}$, $n \in \mathbb{N}$, and inverter I we associate

```
Experiment $\mathbf{Exp}^{\text{n-cp}}_{\mathcal{F},I}(k)$:
For $i = 1$ to $n$ do $(f_i, f_i^{-1}) \xleftarrow{\$} \mathcal{F}$
$(x_1, \ldots, x_n) \xleftarrow{\$} I_1(1^k)$
$(x'_1, \ldots, x'_n) \xleftarrow{\$} I_1(f_1, \ldots, f_n, f_1(x_1), \ldots, f_n(x_n))$
If $(x_1, \ldots, x_n) = (x'_1, \ldots, x'_n)$ return 1 else return 0
```

Define the *n-CP advantage* of I against F as

$$\mathbf{Adv}^{\text{n-cp}}_{\mathcal{F},I}(k) = \Pr\left[\, \mathbf{Exp}^{\text{n-cp}}_{\mathcal{F},A}(k) \Rightarrow 1 \,\right]$$

Call a n-CP adversary $I = (I_1, I_2)$ *canonical* if I_1 on input 1^k outputs $(x_1, \ldots, x_n)$ where x_1 is randomly chosen from $\{0,1\}^k$ and $x_1 = \ldots = x_n$.

We note that the above notion is weaker than the extension of one-wayness to vectors of non-uniform inputs we gave in Section 4.2.1. This stems from the fact that a weaker notion is required to realize CCA-secure encryption as compared to deterministic encryption (which encrypts each input separately).

Construction of adaptive TDFs. Let $\mathcal{F}$ be a TDF genertor with range $\{0,1\}^n$ for $n = n(k)$. We construct a new TDF generator $\mathcal{G}$ as follows:

```
Algorithm $\mathcal{G}(1^k)$:
$(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$
For $b \in \{0,1\}$ and $1 \le i \le n$ do: $(f_{b,i}, f_{b,i}^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$
$g \leftarrow (f, (f_{0,1}, f_{1,1}), \ldots, (f_{0,n}, f_{1,n}))$
$g^{-1} \leftarrow (f^{-1}, (f_{0,1}^{-1}, f_{1,1}^{-1}), \ldots, (f_{0,n}^{-1}, f_{1,n}^{-1}))$
Return $(g, g^{-1})$
```

where g on input x is defined as $(y, f_{y[1],1}(x) \ldots, f_{y[n],n}(x))$ where $y = f(x)$, and g^{-1} on input $y \| y_1 \| \ldots \| y_n$ returns $x = f^{-1}(y)$ if $f_{y[i],i}(x) = y_i$ for all $1 \le i \le n$ and $\perp$ otherwise.

Theorem 5.3.1. Let I be an inverter against $\mathcal{G}$. Then there is a canonical $(n+1)$-CP inverter I' against $\mathcal{F}$ such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{atdf}_{\mathcal{G},I}(k) \;=\; \mathbf{Adv}^{\text{n+1-cp}}_{\mathcal{F},I'}(k)\,.$$

Furthermore, the running-time of I' is the time to run I.

Improved constructions. We note that it is possible to make the scheme more efficient by additionally using a target-collision resistant hash function [13]. Then, the "selector" bits $y[1], \ldots, y[n]$ in the construction are replaced with the bits of the *hash* of $f(x)$. We also note that following [72] it is possible to give a construction based on a correlated-product TDF allowing a slightly weaker correlation among the inputs, based on error correcting codes.

Construction of adaptive tag-based TDFs. The above construction of adaptive TDFs can easily be modified to give a construction of adaptive tag-based TDFs as well. The difference is that in the "selector" bits $b_1, \ldots, b_n$ are replaced with the bits $t_1, \ldots, t_n$ of the tag t. Notably, when we apply our construction of CCA-secure PKE from adaptive tag-based TDFs given in Section 5.2) to the resulting adaptive tag-based TDF, we obtain precisely the CCA-secure PKE scheme of [72].

5.3.2 Constructions from Lossy and All-but-One TDFs

Construction of adaptive TDFs. Let $\mathsf{LTDF} = (\mathcal{F}_1, \mathcal{F}'_1)$ be a lossy trapdoor function and let $\mathsf{ABO} = (\mathcal{F}_2, \mathcal{F}'_2)$ be an all-but-one trapdoor function; wlog we assume its branch-space is $\{0,1\}^n$ for $n = n(k)$. Let $T\colon \{0,1\}^* \to\to (\{0,1\}^n \setminus \{0^n\})$ be a hash function. We construct a new trapdoor function generator $\mathcal{G}$ as follows.

```
Algorithm $\mathcal{G}(1^k)$:
$(f_1, f_1^{-1}) \xleftarrow{\$} \mathcal{F}_1(1^k)$
$(f_2, f_2^{-1}) \xleftarrow{\$} \mathcal{F}_2(1^k, 0^n)$
Return $(f_1, f_2), (f_1^{-1}, f_2^{-1})$
```

where g on input x is defined as $f_1(x), f_2(h, x)$ where $h = T(f_1(x))$, and g^{-1} on input $y_1 \| y_2$ returns $x = f_1^{-1}(y_1)$ if $f_2(h, x) = y_2$ where $h = T(y_1)$ and $\perp$ otherwise.

Theorem 5.3.2. Let I be an inverter against $\mathcal{G}$. Then there are distinguishers D_1, D_2 against $\mathsf{LTDF}, \mathsf{ABO}$ respectively and an adversary A against T such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathrm{atdf}}_{\mathcal{G},I}(k) \quad \leq \quad \mathbf{Adv}^{\mathrm{ltdf}}_{\mathsf{LTDF},D_1}(k) + \mathbf{Adv}^{\mathrm{abo}}_{\mathsf{ABO},D_2}(k) + \mathbf{Adv}^{\mathrm{tcr}}_{T,A}(k) + 2^{-n+s_1+s_2}$$

where s_1, s_2 are the residual leakages of $\mathsf{LTDF}, \mathsf{ABO}$ respectively. Furthermore, the running-times of D_1, D_2, A are the time to run I.

Proof. The games for the proof, which follows [68], are given in Figure 12. We omit to give the code for the constructed adversaries, since they mostly just repeat the code of the relevant games. Equation 30 follows from the following sequence of inequalities:

$$\begin{aligned}
\mathbf{Adv}^{\text{adaptiveTDF}}_{\mathcal{G},I}(k) &= \Pr\left[\, G_1^I \Rightarrow x \,\right] \\
&\leq \Pr\left[\, G_2^I \Rightarrow x \,\right] + \mathbf{Adv}^{\text{abo}}_{\mathsf{ABO},D_2}(k) \\
&\leq \Pr\left[\, G_3^I \Rightarrow x \,\right] + \Pr[G_2^{A_2} \text{ sets } \mathsf{bad}\,] + \mathbf{Adv}^{\text{abo}}_{\mathsf{ABO},D_2}(k) \\
&\leq \Pr\left[\, G_4^I \Rightarrow x \,\right] + \mathbf{Adv}^{\text{tcr}}_{T,A}(k) + \mathbf{Adv}^{\text{abo}}_{\mathsf{ABO},D_2}(k) \\
&\leq \Pr\left[\, G_5^I \Rightarrow x \,\right] + \mathbf{Adv}^{\text{tcr}}_{T,A}(k) + \mathbf{Adv}^{\text{abo}}_{\mathsf{ABO},D_2}(k) \\
&\quad + \mathbf{Adv}^{\text{ltdf}}_{\mathsf{LTDF},D_1}(k) \\
&\leq 2^{-n+s_1+s_2} + \mathbf{Adv}^{\text{tcr}}_{T,A}(k) + \mathbf{Adv}^{\text{abo}}_{\mathsf{ABO},D_2}(k) + \\
&\quad + \mathbf{Adv}^{\text{ltdf}}_{\mathsf{LTDF},D_1}(k)\ .
\end{aligned}$$

□

In fact, as we show in Section 5.5, the construction actually achieves a stronger security property that we call "adaptive lossiness." This is in particular useful for construction CCA-secure deterministic encryption in the standard model.

Construction of adaptive tag-based TDFs. Similarly to our construction of adaptive TDF from correlated-product TDF, the above construction can be adapted to construct a tag-based adaptive TDF instead. The difference is that in the evaluation algorithm, instead of evaluating the all-but-one TDF at branch $T(y_1)$, it is evaluated at branch t, where the latter is the input tag. As before, when we apply our general construction of CCA-secure PKE from TB-ADTFs given in Section 5.2 to the resulting adaptive tag-based TDF, we obtain precisely the CCA-secure encryption scheme of [68].

5.4 On the Complexity of Adaptive Trapoor Functions

In this section, we further study the complexity of adaptive TDFs. First, we show that adaptive TDFs and adaptive tag-based TDFs are strictly *weaker* than correlated-product TDFs, in a black-box sense. Combined with the above results, this means

```
procedure Initialize$G_1$
$(f_1, f_1^{-1}) \xleftarrow{\$} \mathcal{F}_1(1^k)$
$(f_2, f_2^{-1}) \xleftarrow{\$} \mathcal{F}_2(1^k, 0^n)$
$x \xleftarrow{\$} \{0,1\}^k$
$y_1 \leftarrow f(x)$ ; $h^* \leftarrow T(y_1)$ $y_2 \leftarrow f_2(h^*, x)$
Return $(f_1, f_2), y_1 \| y_2$
```

```
procedure Initialize$G_2$ – $G_4$
$(f_1, f_1^{-1}) \xleftarrow{\$} \mathcal{F}_1(1^k)$
$x \xleftarrow{\$} \{0,1\}^k$
$y_1 \leftarrow f(x)$
$(f_2, f_2^{-1}) \xleftarrow{\$} \mathcal{F}_2(1^k, y_1)$
$h^* \leftarrow T(y_1)$
$y_2 \leftarrow f_2(h^*, x)$
Return $(f_1, f_2), y_1 \| y_2$
```

```
procedure Initialize$G_2$ – $G_4$
$f_1 \xleftarrow{\$} \mathcal{F}_1'(1^k)$
$x \xleftarrow{\$} \{0,1\}^k$
$y_1 \leftarrow f(x)$
$(f_2, f_2^{-1}) \xleftarrow{\$} \mathcal{F}_2(1^k, y_1)$
$y_2 \leftarrow f_2(T(y_1), x)$
Return $(f_1, f_2), y_1 \| y_2$
```

```
On inversion query $y$ $G_1$, $\boxed{G_2}$
$y_1 \| y_2 \leftarrow y$
If $T(y_1) = h^*$ then
    bad $\leftarrow$ true ; $\boxed{\text{Return } \perp}$
$x \leftarrow f^{-1}(y_1)$
If $f_2(T(y_1), x) = y_2$ then return $x$
Else return $\perp$
```

```
On inversion query $y$ $G_1$, $\boxed{G_2}$
$y_1 \| y_2 \leftarrow y$
If $T(y_1) = h^*$ then return $\perp$
$x \leftarrow f_2^{-1}(T(y_1), y_2)$
If $f_1(x) = y_1$ then return $x$
Else return $\perp$
```

```
procedure Finalize$(x')$ All Games
Return $x'$
```

Figure 12: Games for the proof of Theorem 4.3.4.

that adaptivity is currently the weakest security property of TDFs known to imply black-box chosen-ciphertext security. We then show that adaptive tag-based TDFs can be realized from an assumption on RSA inversion not known to imply correlated-product TDFs. This further demonstrates the usefulness of our notions and leads to a very efficient chosen-ciphertext secure RSA-based encryption scheme without random oracles (though based on a non-standard assumption).

5.4.1 A Black-Box Separation from Correlated-Product TDFs

Informally, we call a construction of a primitive P_1 from another primitive P_2 is *black-box* if (1) the algorithms of P_1 only access those of P_2 as oracles, and (2) there is an adversary A such that for every adversary B breaking P_2 then A given oracle access to B. See [70, Definition 3] for the formal definition, which will not be important here (the latter calls this "fully black-box," which is the only notion we will be concerned with here).

Very recently, Vahlis [76] showed that there is no black-box construction of CP-TDFs from OW-TDFs. We observe here that his proof in fact extends to rule out a black-box construction of CP-TDFs from adaptive TDFs or adaptive tag-based TDFs as well.

Theorem 5.4.1. There is no black-box construction of correlated-product TDFs from adaptive TDFs or adaptive tag-based TDFs.

The theorem actually follows by extending Vahlis's proof to rule out a black-box construction of correlated-product TDFs from *exponentially-hard* adaptive TDFs. As shown in Section 5.4.2, adaptive tag-based TDFs are implied by exponentially-hard TDFs, so this rules out a black-box construction of correlated-product TDFs from adaptive tag-based TDFs as well. Since Vahlis's proof is rather technical we avoid explaining its details here. Instead, we describe the high-level ideas and point out a minor change needed to give our claimed result.

Similar to most black-box separation results, in order to show that there is no black-box construction of primitive P_1 from primitive P_2, the proof starts by defining an *ideal oracle* O (the ideal version of P_2), and a *break oracle* B. One then shows that (1) there exist an adversary A that breaks any construction of P_1, with the help of a polynomial number of queries to B and (2) P_2 can be securely realized using the ideal oracle O, even when the adversary is given access to B.

Oracle O. Roughly speaking, O is defined as a triple of functions (g, e, d) sampled uniformly at random from the set of all functions with the following property: g maps trapdoors to public keys; $e(pk, \cdot)$ is an independent permutation for every public key pk, and $d(sk, \cdot)$ inverts $e(pk, \cdot)$ if sk is the trapdoor corresponding to pk.

It is easy to see that oracle O constitutes an adaptive TDF; in fact, it is an *exponentially-hard* adaptive TDF. However, as pointed out in [76], O is also correlation secure as the permutations for every public key is chosen independently and uniformly at random.

oracle B. Oracle B is specially designed to break the security of a correlated-product TDF. It takes as input a triple of circuits (G^O, E^O, D^O) which are candidates for a correlation secure TDF, two public keys pk_1, pk_2 and the values $E(pk_1, x)$ and $E(pk_2, x)$. The naive solution would be to let oracle B return x. However, this would make oracle B too powerful and would allow an adversary to break the security of any ideal TDF by letting the two public keys be $pk_1 = pk_2$. This problem is solved by requiring that the public keys of O encoded in pk_1 are *distinct* from those encoded in pk_2. An additional problem is caused by the fact that the adversary can make queries that contain invalid public keys, while detecting invalid keys by oracle B can render it too powerful. This issue is resolved by requiring the adversary to provide a partial oracle $O' = (g', e', d')$ that is defined on a small part of the domain of (g, e, d) such that relative to O', pk_1 and pk_2 are valid public keys.

We refer the reader to [76] for a more formal description of oracles O and B. The following (informal) claims proven in [76] complete the argument.

Claim 5.4.2. There exists an adversary that making a small number of queries to oracles O and B that breaks the security of any correlated-product TDF.

Claim 5.4.3. Let $\mathcal{F}$ be the trapdoor function that simply forwards its inputs to O. For any adversary A that makes a small number of queries to oracles B and O, A

does not break security of $\mathcal{F}$.

In [76], latter claim is proven for the case when "security of $\mathcal{F}$" is interpreted as one-wayness. However, the proof easily extends to the case of adaptivity. Particularly, the bulk of the proof consists of describing a simulator S that simulates the answers for queries made to oracle B. For consistency purposes, S keeps a list O^* of all the query/answers made to the challenge function $e(pk^*, \cdot)$ where pk^* is the challenge public key. In case of adaptive TDFs, S needs to *do the same for any query* $e^{-1}(pk^*, \cdot)$ *made to the inversion oracle*. The rest of the proof stays the same.

5.4.2 Adaptivity versus Tag-Based Adaptivity

Note that tag-based TDFs can be viewed as a specific type of TDF in which the first part of the input is output in the clear. Based on this observation we show that adaptive TDFs and tag-based adaptive TDFs are equivalent under *exponential hardness*, meaning that if we start with an exponentially-hard version of one primitive it implies an exponentially-hard version of the other. Whether the notions are equivalent in general remains open.

Adaptive TDFs from tag-based adaptive TDFs. Let $\mathcal{G}$ be a tag-based adaptive TDF generator with tag-space $\{0,1\}^\ell$ for $\ell = \ell(k)$. Let $T : \{0,1\}^k \to \{0,1\}^\ell$ be a (compressing) TCR hash function. We construct an adaptive TDF generator $\mathcal{G}[T]$ that on input 1^k outputs $g, g^{-1}) \xleftarrow{\$} \mathcal{G}(1^k)$. Evaluation $g(x)$ is defined as $(T(x), g(T(x), x))$, and inversion $g^{-1}(h, y)$ is defined as $g^{-1}(h, y)$.

Theorem 5.4.4. Let I be an inverter against $\mathcal{G}[T]$ defined above. Then there is an adversary A against T and an inverter I' against $\mathcal{G}$ such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathrm{aowf}}_{\mathcal{G}[T],I}(k) \;\leq\; 2^\ell \cdot \mathbf{Adv}^{\mathrm{tb\text{-}aowf}}_{I'}(k) + \mathbf{Adv}^{\mathrm{tcr}}_{T,A}(k) \; .$$

Furthermore, the running-times of I', A are the time to run I.

The idea for the proof is that the inverter can just "guess" the hash value $T(x)$ since $\mathcal{G}$ is exponentially hard.

Tag-based adaptive TDFs from adaptive TDFs. Give an adaptive TDF $\mathcal{F}$, we construct a tag-based adaptive TDF $\mathcal{G}$ with with domain $\{0,1\}^{k-\ell}$ and $\{0,1\}^{\ell}$, where $\mathcal{G}$ on input 1^k returns $(g = f, g^{-1} = f^{-1})$ where $(f, f^{-1}) \xleftarrow{\$} \mathcal{F}(1^k)$. Evaluation $g(t, x)$ is defined as $f(t\|x)$ and inversion $g^{-1}(t, y)$ computes $z \leftarrow f^{-1}(y)$, parses $t\|x \leftarrow z$, and returns x if $t = t'$ and otherwise $\perp$.

Theorem 5.4.5. Let I be an inverter against $\mathcal{G}$ defined above. an inverter I' against $\mathcal{F}$ such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{tb-aowf}}_{\mathcal{G},I}(k) \;\; \leq \;\; 2^{\ell} \cdot \mathbf{Adv}^{\text{aowf}}_{I'}(k) \; .$$

Furthermore, the running-time of I' is the time to run I.

Note that it is not hard to see that in the above construction if $\ell = O(\log k)$, we no longer need the assumption that the underlying adaptive TDF is exponentially hard. In other words adaptive TDFs imply tag-based adaptive TDFs with $(\log k)$-bit tags. However, $(\log k)$-bit tag-based adaptive TDFs seem to be a significantly weaker primitive. Indeed, they can be based on any one-way TDF.[1]

5.4.3 Adaptivity from an Assumption on RSA Inversion

To further demonstrate the usefulness of our new notions, we show that adaptive tag-based TDFs are realizable from an assumption on RSA inversion not known to imply a correlated-product TDF.

Instance-independent RSA. The instance-independent RSA assumption, introduced by Pallier and Villar [66], asserts the difficulty of solving the RSA problem —

[1]To see this, consider the construction where k functions $f_1, \ldots, f_k$ are sampled from a family of one-way TDFs. The tag-based adaptive TDF is defined such that on input x and tag t, $f_t(x)$ is returned; inversion is defined naturally.

that is, computing e-th roots modulo $N = pq$ — even if given access to an oracle that computes e'-th roots modulo N for $e' \neq e$. Of course, due to the homomorphic property of RSA some additional restriction on the exponents is necessary for this to hold; in what follows we require that $e \neq e'$ are primes. Let primes_n denote the set of all n-bit primes and To an inverter I with access to an oracle and $n = n(k)$ we associate

```
$\mathbf{Experiment}\ \mathbf{Exp}_I^{\text{ii-rsa-n}}(k)$:
$p, q \xleftarrow{\$} \mathsf{primes}_{k/2}$ ; $N \xleftarrow{\$} pq$
$e \xleftarrow{\$} \mathsf{primes}_n$
$x \xleftarrow{\$} \mathbb{Z}_N$ ; $y \leftarrow x^e \bmod N$
$x' \xleftarrow{\$} I^{\mathcal{O}_{p,q}(\cdot,\cdot)}(N, y)$
If $x = x'$ return 1 else return 0
```

where oracle $\mathcal{O}_{p,q}$ on inputs y', e' returns $y'^{d'} modN$ where $e'd' = 1 \bmod \phi(N)$ if $e \neq e' \in \mathsf{primes}_n$ and $\perp$ otherwise. Define the *II-RSA advantage* of I for n as

$$\mathbf{Adv}_I^{\text{ii-rsa-n}}(k) = \Pr\left[\,\mathbf{Exp}_I^{\text{ii-rsa-n}}(k) \Rightarrow 1\,\right] .$$

We note that Paillier and Villar [66] used this assumption to show that RSA-based schemes cannot be proven secure in the standard model. More recently, Chevallier-Mames and Joye [26] observed that II-RSA can be used to prove security of encryption schemes as well. We note that [66] actually considered the assumption parameterized by a *fixed* "challenge" e (e.g., $e = 3$). We follow the formulation of [26] and choose e at random from the set of all primes of a given length.

Prime Sequence Generator. Our construction uses the "prime sequence generator" of [20], which for any $n \in \mathbb{N}$ with $k \geq (n+1)/2$ probabilistically constructs an efficiently computable, (with high probability) injective map $\mathsf{phash}_n \colon \{0,1\}^k \to \mathsf{primes}_n$. Namley, one first chooses a random $2(n+1)^2$-wise-independent function $Q \colon \{0,1\}^k \times \{1, \ldots, 2(n+1)^2\} \to \{0,1\}^n$ using the standard polynomial evaluation construct over $\mathbb{F}_{2^{\kappa+1}}$. Then for $t \in \{0,1\}^k$, we define $\mathsf{phash}_n(t)$ to be the first prime in the sequence $Q(t,1), \ldots, Q(t, 2(n+1)^2)$.

Tag-based adaptive TDF from II-RSA. Let phash_n be as defined above for $k \geq (n+1)/2$. We construct a tag-based adaptive TDF $\mathcal{F}[\mathsf{phash}_n]$ with tag-space $\{0,1\}^k$ that on input 1^k outputs RSA parameters $(N, (p,q)$. Evaluation on tag $t \in \{0,1\}^k$ input $x \in \mathbb{Z}_N$ is defined as $x^{\mathsf{phash}}t \bmod N$ and inversion is defined accordingly.

Theorem 5.4.6. Let I be an inverter against $\mathcal{F}[\mathsf{phash}_n]$. Then there is an inverter I' such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{tb-aowf}}_{\mathcal{F}[\mathsf{phash}_n],I}(k) \quad \leq \quad \mathbf{Adv}^{\text{ii-rsa-n}}_{I}(k) + 2^{-\Omega(n)} \; .$$

Furthermore, the running-time of I' is the time to run I.

We stress that the use of the "prime sequence generator" in the construction does not introduce any unproven assumption.

Proof. (Sketch.) We consider two games, which we call G_1 and G_2. Game G_1 is just the adaptive tag-based TDF experiment with I against $\mathcal{F}[\mathsf{phash}_n]$. For Game G_2, we modify the inversion oracle to return $\perp$ whenever I makes an inversion query on a tag t' such that $\mathsf{phash}_n(t') = \mathsf{phash}_n(t)$, where t is the challenge tag.

First, we claim that $\Pr\left[\, I^{G_1} \Rightarrow x \,\right] - \Pr\left[\, I^{G_2} \Rightarrow x \,\right] \leq 2^{-\Omega(k)}$. This follows from the analysis of the prime sequence generator in [20], who show that with probability at least $1 - 2^{-\Omega(n)}$ over the choice of Q in its construction, the set $\{\mathsf{phash}_n(t) \;:\; t \in \{0,1\}^k\}$ contains 2^k *random* and *distinct* n-bit primes.

Next, we claim that we can construct an inverter I' such that $\mathbf{Adv}^{\text{ii-rsa-n}}_{I'} = \Pr\left[\, I^{G_2} \Rightarrow x \,\right]$, which completes the proof. Note that I' receives its challenge exponent e "from the outside," so we need a way of "programming" the prime sequence generator at a given point. For this we can use the ideas of [58], who show that for any $t^* \in \{0,1\}^n$ and random $e^* \in \mathsf{primes}_n$, it is possible to construct the polynomial

$Q = Q_{t^*,e^*}$ for the prime sequence generator in such a way that $\mathsf{phash}_n(t^*) = e^*$ and that for every t_0^*, t_1^*, the distribution of these Q's are $2^{-\Omega(n)}$-close. $\square$

An efficient CCA-secure RSA-based encryption scheme. The above construction of adaptive tag-based TDP leads to a very efficient CCA-secure RSA-based encryption scheme in the standard model. Namely, we apply the "optimized" construction of CCA-secure encryption from adaptive tag-based TDFs given in Section 5.2, based on the transform of [17]. We note that to extract enough hardcore bits from only one application of RSA in the construction we can combine II-RSA with the "small-solutions" RSA problem of [75]. Furthermore, by strengthening II-RSA to allow e, e' to be composites such that $\gcd(e, e') = 1$ and quantifying over *all* e in the assumption, we can "heuristically" use a cryptographic hash function with 512-bit output in place of the prime sequence generator for 80-bit security. The resulting scheme has ciphertexts containing only one group element and, assuming the strengthening to II-RSA discussed above, its encryption time is dominated by one 512-bit exponentiation. In terms of applicability, however, it is unclear if such a standard-model scheme secure based on an interactive assumption about RSA is preferable to a random-oracle scheme based on its one-wayness (such as RSA-OAEP [12]).

5.5 Chosen-Ciphertext Secure Deterministic Encryption

In this section, we show how to build on our previous results to achieve chosen-ciphertext secure deterministic encryption. The latter is a strong notion for trapdoor functions that combines all the strengthenings to one-wayness considered in this thesis.

5.5.1 Constructions in the Random Oracle Model

In the random oracle model, one-way TDFs and adaptive TDFs are equivalent. In fact, we can construct DET-CCA secure deterministic encryption from any IND-CPA randomized encryption scheme that meets a minor extra condition. Namely, the Encrypt-with-Hash deterministic encryption scheme EwHash from Subsection 3.3.1 is DET-CCA secure even if the starting encryption scheme is only IND-CPA and morover no ciphertext occurs with too high a probability. More precisely, the *max-ciphertext probability* $\mathsf{mc}_\Pi(\cdot)$ of encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined as follows: for all $k \in \mathbb{N}$ we let $\mathsf{mc}_\Pi(k)$ be the maximum taken over all $y \in \{0,1\}^*$ and all $x \in \mathrm{PtSp}(k)$ of the quantity

$$\Pr\Big[\,(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)\,;\; c \xleftarrow{\$} \mathcal{E}(pk, x) \;:\; c = y\,\Big]\;.$$

Then Theorem 3.3.1 extends as follows.

Theorem 5.5.1. Let $A = (A_1, A_2)$ be a PRIV-CCA against $\mathsf{EwHash}[\Pi]$ with min-entropy μ, which outputs vectors of size v with messages of length n and makes at most q_h queries to its hash oracle and at most q_d queries to its decryption oracle. Then there exists an IND-CPA adversary B against Π such that for every $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{det-cca}}_{\mathsf{EwHash}[\Pi],A}(k) \;\;\le\;\; \mathbf{Adv}^{\text{ind-cpa}}_{\Pi,B}(k) + \frac{2q_\mathrm{h}v}{2^\mu} + 2q_\mathrm{d}\mathsf{mc}_\Pi(k)\;. \tag{30}$$

Furthermore, the running-time of B is at most that of A plus $O(vn + q_\mathrm{h}T_\mathcal{E})$, where $T_\mathcal{E}$ is the time for a computation of $\mathcal{E}$ on messages of length n.

The requirement that $\mathsf{mc}_\Pi(\cdot)$ be small is quite mild. Most practical encryption schemes have negligible max-ciphertext probability.

Proof. The proof is an extension of the proof of Theorem 3.3.1. In particular, we extend the game-chain there and add an extra proedure to the games and to the code of the constructed IND-CPA adversary B to respond to A's decryption queries. The games for the proof are given in Figure 14 and adversary B is given in Figure 13. Equation (30) follows from the following sequence of inequalities, which we will justify below:

$$\Pr\left[\, G_1^{A_2} \Rightarrow b \,\right] \;\le\; \Pr\left[\, G_5^{A_2} \Rightarrow b \,\right] + \frac{vq}{2^{-\mu}} \tag{31}$$

$$\le\; \Pr\left[\, G_6^{A_2} \Rightarrow b \,\right] + \frac{vq}{2^{-\mu}} + \Pr\left[\, G_5^{A_2} \text{ sets } \mathsf{bad} \,\right] \tag{32}$$

$$\le\; \Pr\left[\, G_6^{A_2} \Rightarrow b \,\right] + \frac{vq}{2^{-\mu}} + q_{\mathrm{d}}\mathsf{mc}_{\Pi}(k)\;. \tag{33}$$

Above, Equation 32 is justify exactly as in the proof of Theorem 3.3.1. Then, Lemma 2.4.1 applies to justify (33). Note that when executed in Game G_5, the probability that a decryption query c made by A_2 is a valid ciphertext (i.e., that does not decrypt to $\perp$) for some message x such that A_2 has not queried x to its hash oracle is at most mc_{Π}. This is because, without any information about $H[x]$, $H[x]$ and the coins used by $\mathcal{E}(pk, x)$ have the same distribution from the perspective of A_2 (namely uniformly random). This implies (33). Now Equation 30 follows by multiplying by 2 and subtracting 1, taking into account the definition of the advantages of A, B.

Finally, to justify the claim about the running-time of B, recall the convention to include in the running-time of A that of its overlying experiment. So, in addition to the time to run A, B's time-complexity is dominated by the time needed to create a hash table containing the elements of $\mathbf{x}_0, \mathbf{x}_1$, which is $O(vn)$, as well a for encrypting each hash query. So its additional overhead is $O(vn + q_{\mathrm{h}}T_{\mathcal{E}})$ as desired. $\square$

5.5.2 Constructions without Random Oracles

We build on the construction of adaptive TDFs in Section 5.3.2 to construct CCA-secure deterministic encryption without random oracles. Recall from Section 3.1 that CCA-secure deterministic encryption can be viewed as a strengthening to adaptive TDFs that also hides partial information about high-entropy inputs. Towards acheiving this notion we show the construction in Section 5.3.2 actually achieves a stronger notion we may call "adaptive lossiness."

```
**Adversary** $B_1(pk)$:
$(\mathbf{x}_0, t_0), (\mathbf{x}_1, t_1) \xleftarrow{\$} A_1(1^k)$
$state \leftarrow \mathbf{x}_0 \| t_0 \| \mathbf{x}_1 \| t_1$
Return $(\mathbf{x}_0, \mathbf{x}_1, state)$
```

```
**Adversary** $B_2(pk, \mathbf{c}, state)$:
$\mathbf{x}_0 \| t_0 \| \mathbf{x}_1 \| t_1 \leftarrow state$
Run $A_2$ on inputs $pk, \mathbf{c}$:
    **On hash query** $pk \| x$ do:
    If $H[x]$ is undefined then
        $H[x] \xleftarrow{\$} \mathsf{Coins}_{pk}(|x|)$
        $E[x] \leftarrow \mathcal{E}(pk, x; H[x])$
        If $x \in \mathbf{x}_0$ then
            If $\mathsf{one} = \mathsf{false}$ then $\mathsf{zer} \leftarrow \mathsf{true}$
        If $x \in \mathbf{x}_1$ then
            If $\mathsf{zer} = \mathsf{false}$ then $\mathsf{one} \leftarrow \mathsf{true}$
    Return $H[x]$
Let $g$ be the output of $A_2$
If $\mathsf{zer} = \mathsf{true}$ then $d \leftarrow 0$
Else If $\mathsf{one} = \mathsf{true}$ then $d \leftarrow 1$
      Else If $g = t_1$ then $d \leftarrow 1$ else $d \leftarrow 0$
Return $d$
```

Figure 13: IND-CPA adversary B for proof of Theorem 5.5.1.

Adaptive lossiness. An *adaptive lossy TDF generator* $\mathsf{ALTDF} = (\mathcal{F}, \mathcal{F}')$ is a pair of algorithms. Algorithm $\mathcal{F}$ outputs a tuple (f, f^{-1}) where f is the (description of) a function with domain $\{0,1\}^k$ and f^{-1} is the inversion of f. On input of some auxiliary information $x^* \in \{0,1\}^k$, algorithm $\mathcal{F}'(x^*)$ outputs a tuple (f, f^{-1}) where f is the (description of) a function with domain $\{0,1\}^k$, and f^{-1} is another function (*not* necessarily the inverse of f). We require that for all $x^* \in \{0,1\}^k$, given f it is computationally hard for an adversary to distinguish whether (f, f^{-1}) was sampled from $\mathcal{F}$ or from $\mathcal{F}'(x^*)$, even given an inversion oracle for f^{-1}. Formally, to a distinguisher D against ALTDF we associate its *ALTDF advantage* $\mathbf{Adv}^{\mathrm{altdf}}_{\mathsf{ALTDF}}(D)$ defined as the maximum over $x^* \in \{0,1\}^k$ of

$$\Pr\left[D^{f^{-1}}(f, x^*) \Rightarrow 1 \ : \ (f, f^{-1}) \xleftarrow{\$} \mathcal{F} \right] - \Pr\left[D^{f^{-1}}(f, x^*) \Rightarrow 1 \ : \ (f, f^{-1}) \xleftarrow{\$} \mathcal{F}'(x^*) \right] .$$

We say ALTDF has *residual leakage* s if for all coins c of $\mathcal{F}'$ the function g defined on $\{0,1\}^k$ as $g(x) := f(x)$ where $(f, f^{-1}) \leftarrow \mathcal{F}'(x; c)$ has $|R(g)| \le 2^s$, where $R(g)$

```
On $H$ query $pk\|x$:    $\boxed{G_1\text{–}G_4}$, $G_5$
If $H[x]$ is undefined then
    $H[x] \xleftarrow{\$} \mathsf{Coins}_{pk}(|x|)$
    $E[x] \leftarrow \mathcal{E}(pk, x; H[x])$
If $\exists i$ such that $x = \mathbf{x}_0[i]$ then
    If $\mathsf{one} = \mathsf{false}$ then $\mathsf{zer} \leftarrow \mathsf{true}$
    $\boxed{H[x] \leftarrow R_{0,i}}$
If $\exists i$ such that $x = \mathbf{x}_1[i]$ then
    If $\mathsf{zer} = \mathsf{false}$ then $\mathsf{one} \leftarrow \mathsf{true}$
    $\boxed{H[x] \leftarrow R_{1,i}}$
Return $H[x]$
```

```
On Decryption query $y$
    $\boxed{G_1 - G_5}$, $G_6$
If $\exists z$ such that $E[z] = y$ then return $z$
$x \leftarrow \mathcal{D}(sk, y)$
If $x = \perp$ then return $\perp$
If $H[x]$ is undefined then
    $H[x] \xleftarrow{\$} \mathsf{Coins}_{pk}(|x|)$
    $E[x] \leftarrow \mathcal{E}(pk, x; H[x])$
    If $E[x] = y$ then
        $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{Return } x}$
Return $\perp$
```

Figure 14: Games for the proof of Theorem 5.5.1. Only differences as compared to the games in Figure 4 are shown.

denotes the range of g. The *lossiness* of ALTDF is $k - s$. We say that $\mathcal{F}$ is adaptive lossy if it is overlain by ALTDF.

CCA-Secure deterministic encryption from ALTDFs. It is not too hard to see that the constructions of DET-CPA determinsitic encryption in Section 4.2 based on LTDFs can be "upgraded" to CCA security by swapping the LTDFs for ALTDFs. The analogous theorem statements hold in these cases.

Construction of ALTDFs. Define $\mathsf{ALTDF} = (\mathcal{G}, \mathcal{G}')$ where $\mathcal{G}$ is as defined in Section 5.3.2 and

```
Algorithm $\mathcal{G}'(1^k, x^*)$:
$(f_1, f_1^{-1}) \xleftarrow{\$} \mathcal{F}_1'(1^k)$
$(f_2, f_2^{-1}) \xleftarrow{\$} \mathcal{F}_2(1^k, ^*)$
Return $(f_1, f_2), (f_1^{-1}, f_2^{-1})$
```

Theorem 5.5.2. Let D be an ALTDF distinguisher against ALTDF defined above.

Then there are distinguishers D_1, D_2 against $\mathsf{LTDF}, \mathsf{ABO}$ respectively and an adversary A against T such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\mathrm{altdf}}_{\mathcal{G},D}(k) \;\; \leq \;\; \mathbf{Adv}^{\mathrm{ltdf}}_{\mathsf{LTDF},D_1}(k) + \mathbf{Adv}^{\mathrm{abo}}_{\mathsf{ABO},D_2}(k) + \mathbf{Adv}^{\mathrm{tcr}}_{T,A}(k)\;. \tag{34}$$

Furthermore, the running-times of D_1, D_2, A are the time to run I, and the residual leakage of ALTDF is $s_{\mathsf{LTDF}} + s_{\mathsf{ABO}}$.

Proof. Equation 34 follows from essentially the same analysis as in the proof of Theorem 5.3.2. It remains to show that for all coins c of $\mathcal{G}'$ the function g defined on $\{0,1\}^k$ as $g'(x) = g(x)$ where $(g, g^{-1}) \leftarrow \mathcal{G}'(x; c)$ has range bounded by $2^{s_{\mathsf{LTDF}}+s_{\mathsf{ABO}}}$. Let $c = (c_{\mathsf{LTDF}}, c_{\mathsf{ABO}})$ be fixed but arbitrary such coins. Then the size of the range of f'_{LTDF} output by $\mathcal{F}'_{\mathsf{LTDF}}$ on coins c_{LTDF} is bounded by $2^{s_{\mathsf{LTDF}}}$ by assumption. Now, every fixed $y_{\mathsf{LTDF}} = f'_{\mathsf{LTDF}}(x)$ uniquely determines $t = T(y_{\mathsf{LTDF}})$, and the range of the function $f_{\mathsf{ABO}}(t, \cdot)$ for f_{ABO} output by $\mathcal{F}'_{\mathsf{ABO}}$ on input x and coins c_{ABO} is bounded by $2^{s_{\mathsf{ABO}}}$ by assumption. Therefore the total range of g' is bounded by $2^{s_{\mathsf{LTDF}}+s_{\mathsf{ABO}}}$ as desired. $\square$

REFERENCES

[1] "Tahoe: A secure distributed file system." Internet website; accessed 7 July 2010. `http://tahoe-lafs.org/~warner/tahoe.html`.

[2] Adya, A., Bolosky, W. J., Castro, M., Cermak, G., Chaiken, R., Douceur, J. R., Howell, J., Lorch, J. R., Theimer, M., and Wattenhofer, R., "Farsite: Federated, available, and reliable storage for an incompletely trusted environment," in *OSDI*, 2002.

[3] Alexi, W., Chor, B., Goldreich, O., and Schnorr, C.-P., "Rsa and rabin functions: Certain parts are as hard as the whole," *SIAM J. Comput.*, vol. 17, no. 2, 1988.

[4] Bellare, M., Boldyreva, A., and Micali, S., "Public-key encryption in a multi-user setting: Security proofs and improvements," in *EUROCRYPT*, pp. 259–274, 2000.

[5] Bellare, M., Boldyreva, A., and O'Neill, A., "Deterministic and efficiently searchable encryption," in *CRYPTO*, pp. 535–552, 2007.

[6] Bellare, M., Brakerski, Z., Naor, M., Ristenpart, T., Segev, G., Shacham, H., and Yilek, S., "Hedged public-key encryption: How to protect against bad randomness," in *ASIACRYPT*, pp. 232–249, 2009.

[7] Bellare, M., Fischlin, M., O'Neill, A., and Ristenpart, T., "Deterministic encryption: Definitional equivalences and constructions without random oracles," in *CRYPTO*, pp. 360–378, 2008.

[8] Bellare, M., Halevi, S., Sahai, A., and Vadhan, S. P., “Many-to-one trapdoor functions and their ralation to public-key cryptosystems,” in *CRYPTO*, pp. 283–298, 1998.

[9] Bellare, M., Kohno, T., and Namprempre, C., “Breaking and provably repairing the ssh authenticated encryption scheme: A case study of the encode-then-encrypt-and-mac paradigm,” *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 2, pp. 206–241, 2004.

[10] Bellare, M., Namprempre, C., Pointcheval, D., and Semanko, M., “The one-more-rsa-inversion problems and the security of chaum’s blind signature scheme,” *J. Cryptology*, vol. 16, no. 3, pp. 185–215, 2003.

[11] Bellare, M. and Rogaway, P., “Random oracles are practical: A paradigm for designing efficient protocols,” in *ACM Conference on Computer and Communications Security*, pp. 62–73, 1993.

[12] Bellare, M. and Rogaway, P., “Optimal asymmetric encryption,” in *EUROCRYPT*, pp. 92–111, 1994.

[13] Bellare, M. and Rogaway, P., “Collision-resistant hashing: Towards making uowhfs practical,” in *CRYPTO*, pp. 470–484, 1997.

[14] Bellare, M. and Rogaway, P., “The security of triple encryption and a framework for code-based game-playing proofs,” in *EUROCRYPT*, pp. 409–426, 2006.

[15] Blum, M. and Micali, S., “How to generate cryptographically strong sequences of pseudo-random bits,” *SIAM J. Comput.*, vol. 13, no. 4, pp. 850–864, 1984.

[16] Boldyreva, A., Fehr, S., and O'Neill, A., "On notions of security for deterministic encryption, and efficient constructions without random oracles," in *CRYPTO*, pp. 335–359, 2008.

[17] Boneh, D., Canetti, R., Halevi, S., and Katz, J., "Chosen-ciphertext security from identity-based encryption," *SIAM J. Comput.*, vol. 36, no. 5, pp. 1301–1328, 2007.

[18] Boneh, D., Crescenzo, G. D., Ostrovsky, R., and Persiano, G., "Public key encryption with keyword search," in *EUROCRYPT*, pp. 506–522, 2004.

[19] Boyen, X. and Waters, B., "Shrinking the keys of discrete-log-type lossy trapdoor functions," in *ACNS*, pp. 35–52, 2010.

[20] Cachin, C., Micali, S., and Stadler, M., "Computationally private information retrieval with polylogarithmic communication," in *EUROCRYPT*, pp. 402–414, 1999.

[21] Canetti, R., "Towards realizing random oracles: Hash functions that hide all partial information," in *CRYPTO*, pp. 455–469, 1997.

[22] Canetti, R. and Dakdouk, R. R., "Towards a theory of extractable functions," in *TCC*, pp. 595–613, 2009.

[23] Canetti, R., Goldreich, O., and Halevi, S., "The random oracle methodology, revisited," *J. ACM*, vol. 51, no. 4, pp. 557–594, 2004. Preliminary version in STOC 1998.

[24] Canetti, R., Halevi, S., and Katz, J., "A forward-secure public-key encryption scheme," *J. Cryptology*, vol. 20, no. 3, pp. 265–294, 2007. Preliminary version in EUROCRYPT 2003.

[25] Canetti, R., Micciancio, D., and Reingold, O., "Perfectly one-way probabilistic hash functions (preliminary version)," in *STOC*, pp. 131–140, 1998.

[26] Chevallier-Mames, B. and Joye, M., "Chosen-ciphertext secure rsa-type cryptosystems," in *ProvSec*, pp. 32–46, 2009.

[27] Chor, B. and Goldreich, O., "Unbiased bits from sources of weak randomness and probabilistic communication complexity," *SIAM J. Comput.*, vol. 17, no. 2, 1988.

[28] Cramer, R. and Shoup, V., "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," in *CRYPTO*, pp. 13–25, 1998.

[29] Damgård, I. and Jurik, M., "A generalisation, a simplification and some applications of paillier's probabilistic public-key system," in *Public Key Cryptography*, pp. 119–136, 2001.

[30] Damgård, I. and Nielsen, J. B., "Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor," in *CRYPTO*, pp. 581–596, 2002.

[31] Damgård, I. and Nielsen, J. B., "Universally composable efficient multiparty computation from threshold homomorphic encryption," in *CRYPTO*, pp. 247–264, 2003.

[32] Dent, A. W., Fischlin, M., Manulis, M., Stam, M., and Schröder, D., "Confidential signatures and deterministic signcryption," in *Public Key Cryptography*, pp. 462–479, 2010.

[33] Desrosiers, S. P., "Entropic security in quantum cryptography," *Quantum Information Processing*, vol. 8, no. 4, pp. 331–345, 2009.

[34] Diffie, W. and Hellman, M. E., "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[35] Dodis, Y., Gennaro, R., Håstad, J., Krawczyk, H., and Rabin, T., "Randomness extraction and key derivation using the cbc, cascade and hmac modes," in *CRYPTO*, pp. 494–510, 2004.

[36] Dodis, Y., Ostrovsky, R., Reyzin, L., and Smith, A., "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," *SIAM J. Comput.*, vol. 38, no. 1, pp. 97–139, 2008.

[37] Dodis, Y. and Smith, A., "Correcting errors without leaking partial information," in *STOC*, pp. 654–663, 2005.

[38] Dodis, Y. and Smith, A., "Entropic security and the encryption of high entropy messages," in *TCC*, pp. 556–577, 2005.

[39] Dolev, D., Dwork, C., and Naor, M., "Nonmalleable cryptography," *SIAM J. Comput.*, vol. 30, no. 2, pp. 391–437, 2000.

[40] Faust, S., Kiltz, E., Pietrzak, K., and Rothblum, G. N., "Leakage-resilient signatures," in *TCC*, pp. 343–360, 2010.

[41] Fehr, S., "Secure deterministic encryption for high-entropy messages," 2008. Unpubished Manuscript.

[42] Fujisaki, E., Okamoto, T., Pointcheval, D., and Stern, J., "Rsa-oaep is secure under the rsa assumption," *J. Cryptology*, vol. 17, no. 2, pp. 81–104, 2004.

[43] Gamal, T. E., "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, 1985.

[44] Gertner, Y., Malkin, T., and Reingold, O., “On the impossibility of basing trapdoor functions on trapdoor predicates,” in *FOCS*, pp. 126–135, 2001.

[45] Goldreich, O., Goldwasser, S., and Micali, S., “How to construct random functions,” *J. ACM*, vol. 33, no. 4, pp. 792–807, 1986.

[46] Goldreich, O. and Levin, L. A., “A hard-core predicate for all one-way functions,” in *STOC*, pp. 25–32, 1989.

[47] Goldwasser, S. and Micali, S., “Probabilistic encryption,” *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[48] Hacigümüs, H., Iyer, B. R., Li, C., and Mehrotra, S., “Executing sql over encrypted data in the database-service-provider model,” in *SIGMOD Conference*, pp. 216–227, 2002.

[49] Hacigümüs, H., Mehrotra, S., and Iyer, B. R., “Providing database as a service,” in *ICDE*, pp. 29–, 2002.

[50] Håstad, J., Impagliazzo, R., Levin, L. A., and Luby, M., “A pseudorandom generator from any one-way function,” *SIAM J. Comput.*, vol. 28, no. 4, pp. 1364–1396, 1999.

[51] Kantarcioglu, M. and Clifton, C., “Security issues in querying encrypted data,” in *DBSec*, pp. 325–337, 2005.

[52] Kaplan, E., Naor, M., and Reingold, O., “Derandomized constructions of -wise (almost) independent permutations,” *Algorithmica*, vol. 55, no. 1, pp. 113–133, 2009.

[53] Kiltz, E., “Chosen-ciphertext security from tag-based encryption,” in *TCC*, pp. 581–600, 2006.

[54] Kiltz, E., Mohassel, P., and O'Neill, A., "Adaptive trapdoor functions and chosen-ciphertext security," in *EUROCRYPT*, pp. 673–692, 2010.

[55] Kiltz, E., Pietrzak, K., Stam, M., and Yung, M., "A new randomness extraction paradigm for hybrid encryption," in *EUROCRYPT*, pp. 590–609, 2009.

[56] Lipton, R., "How to cheat at mental poker," *SIAM J. Comput.*, vol. 36, no. 5, pp. 1301–1328, 2007.

[57] Luby, M. and Rackoff, C., "How to construct pseudorandom permutations from pseudorandom functions," *SIAM J. Comput.*, vol. 17, no. 2, pp. 373–386, 1988.

[58] Micali, S., Rabin, M. O., and Vadhan, S. P., "Verifiable random functions," in *FOCS*, pp. 120–130, 1999.

[59] Micali, S., Rackoff, C., and Sloan, B., "The notion of security for probabilistic cryptosystems," *SIAM J. Comput.*, vol. 17, no. 2, 1988.

[60] Mol, P. and Yilek, S., "Chosen-ciphertext security from slightly lossy trapdoor functions," in *Public Key Cryptography*, pp. 296–311, 2010.

[61] Myers, S. and Shelat, A., "Bit encryption is complete," in *FOCS*, pp. 607–616, 2009.

[62] Naor, M. and Reingold, O., "On the construction of pseudorandom permutations: Luby-rackoff revisited," *J. Cryptology*, vol. 12, no. 1, pp. 29–66, 1999.

[63] Naor, M. and Yung, M., "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *STOC*, pp. 427–437, 1990.

[64] O'Neill, A., "Deterministic public-key encryption revisited." Cryptology ePrint Archive, Report 2010/533, 2010. `http://eprint.iacr.org/`.

[65] Paillier, P., "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT*, pp. 223–238, 1999.

[66] Paillier, P. and Villar, J. L., "Trading one-wayness against chosen-ciphertext security in factoring-based encryption," in *ASIACRYPT*, pp. 252–266, 2006.

[67] Pandey, O., Pass, R., and Vaikuntanathan, V., "Adaptive one-way functions and applications," in *CRYPTO*, pp. 57–74, 2008.

[68] Peikert, C. and Waters, B., "Lossy trapdoor functions and their applications," in *STOC*, pp. 187–196, 2008.

[69] Rackoff, C. and Simon, D. R., "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack," in *CRYPTO*, pp. 433–444, 1991.

[70] Reingold, O., Trevisan, L., and Vadhan, S. P., "Notions of reducibility between cryptographic primitives," in *TCC*, pp. 1–20, 2004.

[71] Rivest, R. L., Shamir, A., and Adleman, L. M., "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[72] Rosen, A. and Segev, G., "Chosen-ciphertext security via correlated products," in *TCC*, pp. 419–436, 2009.

[73] Russell, A. and Wang, H., "How to fool an unbounded adversary with a short key," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 1130–1140, 2006.

[74] Shoup, V., *A computational introduction to number theory and algebra.* New York, NY, USA: Cambridge University Press, 2005.

[75] Steinfeld, R., Pieprzyk, J., and Wang, H., "On the provable security of an efficient rsa-based pseudorandom generator," in *ASIACRYPT*, pp. 194–209, 2006.

[76] Vahlis, Y., "Two is a crowd? a black-box separation of one-wayness and security under correlated inputs," in *TCC*, pp. 165–182, 2010.

[77] Wee, H., "Efficient chosen-ciphertext security via extractable hash proofs," in *CRYPTO*, 2010.

[78] Yao, A. C.-C., "Theory and applications of trapdoor functions (extended abstract)," in *FOCS*, pp. 80–91, 1982.

[79] Yilek, S., Rescorla, E., Shacham, H., Enright, B., and Savage, S., "When Private Keys are Public: Results from the 2008 Debian OpenSSL Vulnerability," in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement – IMC 2009*, pp. 15–27, ACM, 2009.